Quick BASIC
初歩の初歩講座

興味深いショート・プログラムを豊富に収録。
そのサンプル・プログラムをもとに、プログラムの組み方を初め、
コンパイルやリンカーの使い方を徹底的にわかりやすく解説。

ASIC

b

Quick

遠藤 謙一

Quick BASIC
初歩の初歩講座

遠藤 謙一

ASIC

uick

はじめに

なぜ、今ここにきてQuick BASICが受けるのでしょう。

それは、MS-DOS上で使えるBASICであり、CやPascalのように、コンパイルして実行用ファイルを作成することができるからです。それもQuick BASICの画面上にあるメニューを選択するだけで、わずらわしい操作法を知らなくても、ワンタッチで実行ファイルを作れるということに最大の魅力があります。とりあえず、実行ファイルを作る、それから、徐々にコンパイラやリンカのことを学習しようという人には、うってつけの言語です。それと、プログラムを組むときに、仕様書やフローチャートを書いたり、机上で完全なプログラムを書かなくても、「とりあえず走らせてみよう」とBASIC本来のインタプリタで実行を繰り返し、バグを完全に取り終わってからコンパイルするので失敗もありません。

そして、なによりもうれしいのが、このQuick BASICのシステムを買えば、あとからコンパイラを追加購入しなければならないなどということもありません。この経済性も見逃せません。

Quick BASICは、GOTO文を使ったプログラムでも、CやPascalのようにGOTO文のないプログラムでも記述することができ、過去にBASICを学んだことのある人にとっては、そのままBASICの知識を生かすことができるのも大きな魅力です。

第一部の入門編ではQuick BASICの画面編集のキー操作などは、必要最小限のことだけを記述しています。詳しいことが知りたかったら「ユーザーズマニュアルをお読みなさい」と、ここでは冷たく突き放すことにして、その分サンプルプログラムを詰め込むことにしました。

BASICは「INPUT」と「PRINT」それに、「IF・・・・THEN～ELSE」の3つ

を組み合わせるとかなりのプログラムが組むことができます。できるだけこの3つで記述しようとしたため、やや回りくどいと思われるようなプログラムも多くなっています。サンプルプログラムを入力しながら、「なぜ？」「どうして？」と考えながら、スマートなプログラムに作り変えてください。

プログラムを組むのは“あなた”です。BASICを学ぶのではなくBASICによるプログラムの組み立て方を学習するのだということを知っておいてください。

私たちは、中学校、高校、大学と英語を10年以上学習していながら、実際に役立つ英語力が身につかないと言われています。でも、英語を知らなくても、実際に英語圏で生活するようになると半年で英語が話せるようになるとも言われます。BASICもひとつの言語です。いろいろな本を読んで理屈を覚えるよりも、まず、BASICでプログラムを作り、実行結果を見て、よく考える。これが、もっともオーソドックスなBASICの学習法だと思います。幸いにも、BASICのプログラムが掲載された書籍は、たくさん出版されています。

本書もそうした書籍のひとつです。本書では、ヒントとなりそうな部品としてプログラムを記述しています。完成したプログラムは、そういくつもありません。部品として記述されたプログラムをどう組み立てて実用的なプログラムにするかということは、“あなた”が決めてください。

第二部の応用編は、BASICのもっとも得意とするグラフィックスです。特に計算式を使った円の回転や四角形の移動などは、短いプログラムで楽しむことができます。本書では、「点」と「線」と「円」を組み合わせた「移動」と「回転」を最終的なテーマとして取り組んでいきます。また、「パソコンゲーム」の原点となるカーソル移動キーを使ったプログラムは、かなりしつこく取り上げています。これもヒントとして、どのように、自分のプログラムに生かすかは、やっぱり“あなた”次第でしょう。

Quick BASICではたくさんのサンプルプログラムが用意されています。ぜひ、活用してください。

また、グラフィックスのショートプログラムを掲載した書籍は『パソコン図形処理テクニック』や『パソコン模様構成テクニック』（ともに、誠文堂新光社刊：赤松義幸著）などが出版されています。これらは、N88-BASIC用に書かれたものですが、「グラフィックスをやってみたい」と思う人には必読書といってよいほどの名著です。N88-BASIC用プログラムをMS-DOS上のQuick BASICに移植するのは容易ですから、Quick BASICでも十分に活用できます。

本書では、具体的な移植例として『パソコン図形処理テクニック』に掲載されたいくつかのオリジナル作品を使わさせていただきましたので、参考にしてください。

なお、移植に使用したプログラムは、『パソコン図形処理テクニック』の著者、赤松義幸氏と誠文堂新光社編集部の好意により許可を受けたもので、本書で使用したプログラムおよび、移植したプログラムは、原本著者に著作権があります。本書を執筆するにあたり、筆者の申し入れを快諾してくださった赤松義幸氏、および誠文堂新光社の編集部の方々に誌上をお借りして、厚くお礼申し上げます。

平成２年7月

遠藤　謙一

目　次

はじめに　3

1　入門編

第１章　起動するまでに実行すること、考えること　10

1.1　従来のBASICとの違い　10
1.2　Quick BASICを動かすために用意するもの　12
1.3　Quick BASIC　V4.5に入っているファイル　13

第２章　「QB起動ディスク」を作る　19

1.1　セットアップの前準備　19
2.2　セットアッププログラムの使い方　22
2.3　セットアップユーティリティメニュー　25
2.4　日本語フロントエンドプロセッサの組み込み　29
2.5　ハードディスクへのセットアップ　37
2.6　Quick BASICの起動と終了　41

第３章　プログラムのコンパイル　52

3.1　コンピュータへの命令と結果の表示　52
3.2　入力の繰り返しと分岐　58
3.3　Ｖ４．５での実行用ファイルの作成　66
3.4　Ｖ４．２で独立型実行用ファイルの作成　71
3.5　Ｖ４．２ランタイム分離型実行用ファイルの作成　74
3.6　プログラムの実行速度　78

第4章　画面表示の体裁を整える　85
4.1　セミコロンとカンマ　85
4.2　狙った位置に文字表示　90
4.3　数字と文字の桁合わせ　96
4.4　文字関数　100

第5章　ファイル処理の基礎知識　112
5.1　シーケンシャルファイルの作成　113
5.2　ロータス1-2-3用データ入力プログラム　116
5.3　データ入力プログラムの実行可能(.EXE)ファイルの作成　129
5.4　ロータス1-2-3でデータを使う　132

2 応用編

第1章　図形を描くための基本的な命令　140
1.1　グラフィックスのおまじない　140
1.2　画面に点を表示する　141
1.3　直線の表示　143
1.4　四角形の線と面　146
1.5　円を描く　147
1.6　色をつける　148

第2章　直線と四角形　153
2.1　直線のスタイル　153
2.2　乱数を使った点と線　155
2.3　三角形の作成　158
2.4　乱数を使った円と矩形　162
2.5　PUT、GETでウィンドウを作る　167

第3章　曲　線 170

3.1 CIRCLE文を使った円弧 170
3.2　関数を使った曲線 174
3.3　三角関数を使う 178

第4章　図形の回転 187

4.1　円の回転 187
4.2　直線を使った曲線 191

第5章　N88-BASICからの移植 204

5.1　N88-BASICのプログラムを使う 204
5.2　先輩の資産を生かす 208
5.3　未完の表計算プログラムの移植 214

付　録

1　Quick BASIC V4.2のセットアップ 226
2　マウスドライバの利用方法 239
3　変数名に使うことができないBASICの予約語 241
4　V4.5の制限 242
5　4.2と4.5の違い 243
6　本誌サンプルプログラム内容一覧 244

Quick BASIC

1 入門編

第１章　起動するまでに実行すること、考えること

1.1 従来のBASICとの違い

従来、パソコンを購入すると付属としてついてくるディスクBASICを使うと、画面上に、

```
How many files(0-15)?
NEC N-88 BASIC(86) version xx.xx
 Copyright (c) xxxx by NEC Corporation / Microsoft Corp.
xxxxxx Bytes free
Ok
```

※画面のxxxxには、バージョンによって違う数字が入ります

などと表示され、N88-BASICが起動します。

ここで、

```
10 PRINT 123+456
```

と、プログラムを入力してRUNさせればプログラムは実行され、画面上に579と表示されます。ただ（無料？）でついてくるBASICを使えばいいのに、わざわざQuick BASICを使うのは、なぜでしょう。

その最大の理由は、Quick BASICがMS-DOS上で動いているBASICだからではない

でしょうか。自分で作ったプログラムが“「マルチプラン」や「一太郎」と互換性をもつ”、これは考えただけでわくわくしてきます。

さらに、コンパイルすることで、Quick BASICのシステムがなくても、自分の作ったプログラムをすぐに実行できるEXEファイルやCOMファイルにできることも、Quick BASICを使う大きな理由でしょう。このコンパイルの作業は、メニュー上で簡単に実行できるようになっています。

そして、なによりも、N88-BASICのMS-DOS版インタプリタの値段で、コンパイラまで購入できるという経済的負担の軽さが従来のBASICとの大きなちがいかと思います。

もし、MS-DOS版N88-BASICインタプリタと、別売のコンパイラを購入して、実行用のプログラムを作成するとすると、次のような手順を必要とします。

1. プログラムの仕様を考える
2. エディタと呼ばれるプログラムを起動して、プログラムをコンピュータに打ち込む
3. 打ち込んだプログラムの名前を登録して、エディタを終了させる
4. コンパイルという機械語への翻訳作業プログラムを起動する
5. オブジェクトファイルと呼ばれる名前をつけて（自動的につけてくれるのもある）、コンパイルを終了する
6. リンカを使って、必要なライブラリを結合させて、完全な実行用ファイルに仕上げる
7. 実行用ファイル名をつけてリンカを終了させる

もし、プログラムに入力ミスがあった場合には、1から繰り返すことになります。また、思わぬプログラムの記述のちがいで、実行用プログラムが使えなかったときにも、1からの作業を繰り返すことになります。

ひとつのプログラムを起動して、「エディタ機能」も「コンパイル機能」も「リンカ機能」もあって、プログラムの記述ミスや入力ミスを簡単に修正できるものがないものだろうか……？　ということを誰しも考えることではないでしょうか。これらの複雑な作業を一発でやってくれる環境のことを「統合開発環境」といっています。Quick BASICはこの環境をもっているのです。この「統合開発環境」という

のもQuick BASICの大きな特徴のひとつです。

これで、誰もがコンパイル／リンカを簡単にできるようになり、おなじみのBASICが使えるわけです。

さらに、BASICプログラムの弱点である「GOTO文によるスパゲティー」といわれた、わかりにくいプログラムを書かなくてすむ「構造化プログラミング」という方法が取り入れられました。これもQuick BASICの大きな特徴です。

これで、わかりやすいプログラムを書くことができるようになりました。そのため、C言語やPascal言語などで使われているレコード型といわれるデータの型が強化されたり、いくつかの新しい考え方と言語が追加されています。ただ、構造化できるんだから、なにがなんでもプログラムから「GOTO文」をなくそうなどと、堅苦しく考えないほうがいいでしょう。

Quikc BASICは、従来のGOTO文や行番号を、そのまま使ったプログラムでも実行できます。単に、「GOTO文」を使わないプログラムだから“構造化”ではありません。本書の例文プログラムを入力しているうちに、自然と「GOTO文」を使わなくなります。要は、わかりやすいプログラムを作る方法が強化されたと理解してください。

1.2 Quick BASICを動かすために用意するもの

現在、発売されているQuick BASICは、NECのPC-9801とIBM／AXを対象にしていますが、本書では、NEC 9801シリーズを対象に説明します。

1 ハードウエア

本体：NEC　PC-9801シリーズ

ただし、XL・XL2およびXA・LTのハイレゾリューション・モードは除く。

◎本書では、PC-9801vm2を10MHzクロックで使用。

メモリ：640KB

◎I･OデータのPIO-9234Gシリーズの2MRAMボードを使用して640KBに拡張。

フロッピーディスクドライブ：1MBタイプが望ましい（3.5インチ2HD、5インチ2HD、8インチ2D）。

◎本機では、1MB／640KB自動切り替えドライブを搭載。1MBを使用。

RAMディスク：本体メモリが640KBに満たないときには必要。それ以上は、あれば便利。

◎I･OデータのPIO-9234Gシリーズの2MRAMボードを使用。

ハードディスク：あってもなくてもよいが、あれば便利。

◎Itecの20MBハードディスクを使用。

マウス：あってもなくてもよいが、あれば便利。

◎バスマウスを使用。

2　ソフトウエア

OS：MS-DOS Ver.2.XX以上

◎本書では、MS-DOS Ver.3.3Aを使用。

言語：Quick BASIC Ver.4.5（マスターディスク4枚：セットアップディスクは、3枚になります）

RAMディスクドライバ：RAMディスクがなければ不要。

◎本書では、IOS-10 Ver.2.2を使用。

本体メモリは、最低640KBが必要です。キャッシュディスクやRAMディスクを使うときには、RAMディスクドライバが必要となります。

1.3 Quick BASIC　V4.5に入っているファイル

Quick BASICには『セットアップディスク』、『プログラムディスク』、『ライブラリディスク』、『アドバイザディスク』の4枚のフロッピーディスクが、ハードケースに入っています。それぞれのディスクには、次のようなファイルが収納されています。

1　セットアップディスク

セットアップディスクの中のファイルの一覧です。※マークは、MS-DOSのTYPEコマンドで見ることができるテキストファイルになっています。

PACKING	LST	※	パッキングリスト
QBE	DOC	※	QBE.EXE(エミュレータ版)の説明
README	DOC	※	マニュアルの最新の補足説明
SAMPLE	DOC	※	サンプルプログラムの説明
SETUP	@@@		セットアップファイル
SETUP	EXE		セットアッププログラム
●SAMPLE <ディレクトリ>			
DEMO1	BAS	※	効果音プログラム。行番号を使用した従来のプログラム
DEMO2	BAS	※	DEMO1.BASを部分修正したプログラム
DEMO3	BAS	※	構造化されたQuick BASICスタイルのプログラム
QB	BI	※	Quick BASIC 標準インクルードファイル
QCARDS	BAS	※	Learning to use Quick BASICで記述した簡易データベースプログラムQCARDS
QCARDS	DAT		QCARDS用データ
REMLINE	BAS	※	不要な行番号を取り除くユーティリティ
SORTDEMO	BAS	※	アルゴリズム別ソートプログラム
TORUS	BAS	※	グラフィックのデモ
WALTZ	BAS	※	BGMを流しながら、グラフィックスを描くプログラム
●SAMPLE¥ADVR_EX <ディレクトリ>			
	アドバイザで参照しているサンプルプログラムが20種類入っています。		
●SAMPLE¥EXAMPLE <ディレクトリ>			
CCALL	BA	※	Cで作成されたプロシージャのクイックライブラリを呼び出す例。このプログラムの実行は、[QB / LCCALL]で起動
CCALL	C	※	CCALL.BASから呼ばれるモジュールのソース
CCALL	QLB		CCALL.Cで作成したクイックライブラリ
COLOR	BAS	※	COLORステートメントの動作を確認するプログラム
FONTMAG	BAB	※	文字列の拡大・回転表示プログラム
GNCONST	BI	※	各種定数値を定義したインクルードファイル
GNPROC	BAS	※	汎用プロシージャ。汎用性のあるサブルーチン集

GNPROC　BI　※　汎用プロシージャを宣言したインクルードファイル
HANOI　BAS　※　グラフィックスを使ったパズル、ハノイの塔のプログラム
HWCONST　BI　※　ハード依存の定数値を定義したインクルードファイル
QBL　BAS　※　QBで書かれたプログラムをBC.EXEでコンパイルする
SIEGE　BAS　※　ゲームサンプル
SIEGE　MAK　※　SIEGE.BASで必要なモジュールを定義しているファイル
SIEGE.BASを読み込むと自動的にGNPROC.BASを読み込む
SIERP　BAS　※　シェルピンスキー曲線を描画するプログラム
ULCONV　BAS　※　大文字と小文字を変換するフィルタ

●SAMPLE¥BOOK　<ディレクトリ>

ハンドブックマニュアルに記載されているサンプルプログラム

BALLPSET　BAS　※　GETとPUTを使ったアニメーション　P.241
BALLXOR　BAS　※　PUTステートメントのXORオプションを使用　P.244
BAR　BAS　※　棒グラフの作成　P.246
CAL　BAS　※　万年カレンダー　P.150
CHAP210　BAS　※　CHAINによる他のプログラムの呼び出し　MAIN.BAS　P.87
CHAP210A　BAS　※　入力データの平均　MEAN.BAS　P.89
CHAP210B　BAS　※　入力データの標準偏差　STDEV.BAS　P.88
CHAP2_1　BAS　※　円柱の体積と密度　メインモジュール　P.79
CHAP2_1　MAK　※
CHAP2_1A　BAS　※　再帰FUNCTIONプロシージャを使った階乗の計算　P.85
CHAP2_1B　BAS　※　プロシージャの定義モジュール（円柱）　P.79
CHAP3_1　BAS　※　ファイルコピーユーティリティ　P.129
CHAP5_1　BAS　※　グラフィックス　階段線を引く　P.197
CHAP6_1　BAS　※　エラー処理ルーチン　エラーの識別　P.269
CHAP6_2　BAS　※　ミュージックトラップサブルーチン　P.283
CHAP6_3　BAS　※　エラー処理　メインモジュール　P.290
CHAP6_3　MAK　※
CHAP6_3A　BAS　※　エラー処理　GLOBALHANDLERモジュール　P.291
CHAP6_3B　BAS　※　エラー処理　SHORTSUBモジュール　P.290
CHECK　BAS　※　小切手の残高の管理　P.33
COMMONS　BI　※　CHAP210.BASのインクルードファイル
CRLF　BAS　※　キャリッジリターンとラインフイルドフィルタ　P.35
CUBE　BAS　※　立方体の回転
DISPLAY　BAS　※　デバイスの操作　P.147

EDPAT	BAS	※	パターンエディタ　P.259
FILERR	BAS	※	ファイルアクセスエラーのトラッピング　P.295
GET	BAS	※	GETによるイメージ保存　P.238
INDEX	BAS	※	ランダムアクセスファイルの検索　P.155
MANDEL	BAS	※	フラクタルのグラフィックス　P.253
PALETTE	BAS	※	グラフィックス　パレット
PLOTTER	BAS	※	DRAW:グラフィックスマクロ言語　P.229
QLBDUMP	BAS	※	QBライブラリのプロシージャを表示　P.143/P.458
SINEWAVE	BAS	※	グラフィックスの波形　P.213
STRTONUM	BAS	※	文字列から数値への変換　P.188
STYLE	BAS	※	グラフィックス　ラインスタイル　P.200
TERMINAL	BAS	※	端末エミュレータ　P.164
WHEREIS	BAS	※	再帰的なディレクトリの検索　P.89

*プログラムファイルのうしろについているページはマニュアルのページです。

●LEARN <ディレクトリ>

Quick BASICチュートリアルプログラム

BALL	BAS	※
HELP	HLP	
LEARN	EXE	
MENU	ATB	
MENU	TXT	
PRINT	HLP	

2　プログラムディスク

BINとMOUSEの2つのディレクトリが入っています。BINは、Quick BASIC実行ファイルです。

●BIN <ディレクトリ>

BC	EXE	BASICコンパイラ
BRUN45A	EXE	ランタイムモジュール。BRUN45A.LIB に対応
LIB	EXE	ライブラリマネージャ

LINK	EXE		リンカ
QB	EXE		Quick BASICインタープリタ(代替数値演算)
QB	INI		環境設定保存ファイル

●BIN¥EM <ディレクトリ>……エミュレータ版Quick BASIC

BRUN45E	EXE		ランタイムモジュール。BRUN45E.LIB に対応
QBE	EXE		Quick BASICインタープリタ(エミュレータ演算)

●MOUSE <ディレクトリ>

MOUSE	COM		マウスドライバ(常駐型)
MOUSE	SYS		マウスドライバ(デバイスドライバ)
MOUSE	DOC	※	マウスドライバの説明とマウスファンクション

3　ライブラリディスク

●LIB <ディレクトリ>……ライブラリ

ABSOLUTE	ASM	※	ABSOLUTE ソースファイル
BCOM45A	LIB		独立型実行ファイルを作成するためのライブラリ (代替数値演算ライブラリ)
BQLB45	LIB		Quickライブラリを作成するためのライブラリ
BRUN45A	LIB		分離型実行ファイルを作成するためのライブラリ 実行時にBRUN45A.EXEが必要。代替数値演算ライブラリ
INTRPT	ASM	※	INTERRUPT ソースファイル
QB	QLB		標準Quickライブラリ
QB	LIB		標準ライブラリ

●LIB¥EM <ディレクトリ>……エミュレータ用ライブラリ

BCOM45E	LIB		独立型実行ファイル作成用ライブラリ エミュレータ演算ライブラリ
BRUN45E	LIB		分離型実行ファイルを作成するためのライブラリ 実行時にBRUN45E.EXEが必要。エミュレータライブラリ

●USERLIB <ディレクトリ>……ユーザライブラリ

ALL_MAKE	BAT	※	メイクバッチファイル
DOSCALL	OBJ		汎用ルーチン用 オブジェクトファイル
GEN	BAS	※	汎用ルーチン ライブラリ ソースファイル

GEN	BI	※	汎用ルーチン ライブラリ インクルードファイル
GEN	LIB		汎用ルーチン ライブラリ
GEN	QLB		汎用ルーチン Quick ライブラリ
GENDEMO1	BAS	※	汎用ルーチン ライブラリ サンプル
GENDEMO2	BAS	※	汎用ルーチン ライブラリ サンプル
GEN_MAKE	BAT	※	メイクバッチファイル
GRAPH	BAS	※	拡張グラフィック ライブラリ ソースファイル
GRAPH	QLB		拡張グラフィック Quickライブラリ
GRAPH	BI	※	拡張グラフィック ライブラリ インクルードファイル
GRAPH	LIB		拡張グラフィック ライブラリ
GRPDEMO1	BAS	※	拡張グラフィックス サンプル
GRPDEMO2	BAS	※	拡張グラフィックス サンプル
GRP_MAKE	BAT	※	メイクバッチファイル
MOUSE	BAS	※	マウス ライブラリ ソースファイル
MOUSE	BI	※	マウス ライブラリ インクルードファイル
MOUSE	LIB		マウス ライブラリ
MOUSE	QLB		マウス Quickライブラリ
MOUSEVNT	ASM	※	マウスイベント ソースファイル
MOUSEVNT	OBJ		マウスイベント オブジェクトファイル
MUSDEMO1	BAS	※	マウス サンプル
MUSDEMO2	BAS	※	マウス サンプル
MUS_MAKE	BAT	※	メイクバッチファイル

4 アドバイザディスク

QBADVR <ディレクトリ>……ヘルプファイル

HELPMAKE	EXE		ヘルプ作成ツール
QB45QCK	HLP		
QB45ADVR	HLP		
QB45ENER	HLP		
QB45USER	HLP		
QB45USER	QH	※	QB45USER.HLPのテキストファイル。ヘルプを追加する場合に、このファイルを変更

第2章　「QB起動ディスク」を作る

1.1　セットアップの前準備

「QB起動ディスク」の章でのセットアップの方法は、『フロッピーディスクへのセットアップの方法』と、『ハードディスクへのセットアップ』の方法があります。ここでは、フロッピーディスクへのセットアップで、なおかつ、V4.5について行ないます。ハードディスクについては、「2.5　ハードディスクへのセットアップ」を参照してください。もし、V4.2についての説明が必要であれば、本書の『付録』を参照してください。

セットアップを開始する前に、MS-DOSシステムを転送したフォーマット済みフロッピーディスク1枚と、フォーマットのみのフロッピーディスク2枚を準備する必要があります。

1　ディスクのフォーマット

1枚目のフロッピーディスクにはMS-DOSシステムを転送します。

MS-DOSのシステムを転送するには /S スイッチをつけてフォーマットしてください（Aドライブに MS-DOSのシステムディスク、Bドライブにフォーマットするディ

スクを入れているものと仮定します）。
B:を忘れないように！

```
A>FORMAT/S B: [リターン]
```

```
A>FORMAT B:/S
Format Version 4.10

新しいディスクをドライブ B: に挿入し
どれかキーを押してください

ディスクのタイプは  1 : 640(KB)  2 : 1(MB)  = 2

目的のディスクは 1MB FD です

フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
                                            (%)
システムを転送しました

  1250304 バイト  全ディスク容量
   120832 バイト  システム領域
  1129472 バイト  使用可能ディスク容量

別のディスクをフォーマットしますか(Y/N)
  C1  CU  CA  S1  SU   VOID  NWL  INS  REP  ^Z
```

残りの2枚は、システムの転送は行ないません。

```
A>FORMAT B: [リターン]
```

フォーマットが終了したら、ディスクを入れ替えて、“Y”を押してください。

2 付属ラベルを貼る

フォーマットしたフロッピーディスクに、付属のディスクラベルを貼ります。セットアップは、ディスクの交換をこのラベル名で指示しますので必ず貼るようにしてください。

1．MS-DOSを転送したディスクには、［QB起動ディスク］ラベルを貼ります。

2．残りのディスクには、［QBワークディスク］、［QBアドバイザディスク］のラベルを貼ってください。日本語フロントエンドプロセッサを使う人は、手書きのラベルで、「辞書ディスク」を作ります。2HDのフロッピーディスクを使っているときには、QBコンパイラは、［QB起動ディスク］にセットアップされます。

3　MS-DOSのシステムのない人

MS-DOSシステムをもっていない人は購入してください。――で、終わりというのは、あまりに味気ない。もし、MS-DOSのシステムをもっていない人は、いずれ購入されたほうがいいとは思いますが、とりあえず、『一太郎』か『新松』などのMS-DOSで動くアプリケーションソフトを使うことにします。また、そのほかのアプリケーションソフトを使用する場合でも、FORMAT.EXEあるいはFORMAT.COMのファイルがあれば使うことができます。一太郎の中には、FORMAT.COMが入っています。

1．アプリケーションソフトを起動して、終了させます。
2．画面上に　A>が表示されている状態とします。

これで、以後、MS-DOSのシステムは、アプリケーションソフトから借用することとします。

日本語ワードプロセッサ
一太郎 Ver. 3.0
for NEC PC-9801　with ATOK6
(C)1987 (株)ジャストシステム

2.2 セットアッププログラムの使い方

セットアッププログラムを起動するためには、MS-DOSのシステムを起動させる必要があります。そして、起動されたセットアッププログラムは、画面上にメッセージを表示しますから、じっくりと読んで、画面の指示に従ってください。

このセットアッププログラムは、「Quick BASICの利用環境の作成」と「QBの概要と操作環境の説明」を行ないます。

1 MS-DOSのシステムの起動

フォーマットするときに使ったMS-DOSのシステムを、Aドライブに入れて、リセットボタンを押します。そして、画面をA>の状態にします。

もし、画面がA>の状態になっているなら、MS-DOSのシステムを入れ直す必要はありません。

1．画面にA>が出ていることを確認します。
2．Aドライブに、Quick BASICのマスターディスク「セットアッププログラム」を入れます。
3．SETUP［リターン］

Quick BASICセットアップユーティリティが起動され、画面に、連続4枚のメッセージが表示されます。プログラムを中断する場合には、[CTRL]+[C]を押します。

QuickBASIC　セットアップ　ユーティリティ　Version 1.00

フロッピーディスクにセットアップする方へ　(1/4)

(ハードディスクにセットアップする方は，次のページをご覧ください)
セットアップを開始する前にフロッピーディスクの準備が必要です.
一度，CTRL+Cでセットアッププログラムを中断して以下の処理を行ってください.

1. フォーマット済みのフロッピーディスクを3枚用意します.
   1枚目のフロッピーディスクにはMS-DOSシステムを転送してください.
   (MS-DOS のシステムを転送するには　/S スイッチをつけてフォーマットしてください.)

2. フォーマットしたフロッピーディスクに，付属のディスクラベルを貼ります.
   セットアップは，ディスクの交換をこのラベル名で指示しますので必ず貼るようにしてください. MS-DOSを転送したディスクには，[QB起動ディスク]ラベルを貼ります. 同様に残りのディスクには，[QBワークディスク]，[QBアドバイザディスク]のラベルを貼ってください.

【何かキーを押して下さい】

▲1枚目

QuickBASIC　セットアップ　ユーティリティ　Version 1.00

セットアッププログラムの使い方　(2/4)

このセットアッププログラムは，「QuickBASICの利用環境の作成」と「QBの概要と操作環境の説明」を行います. メニューで処理を選択してください.

●「QuickBASICの利用環境の作成」
セットアッププログラムが，いくつかの質問をします. 各質問についての説明が画面の下部に表示されるので，それを参照して質問に答えてください.

1画面ごとに，その画面の設定で良いかどうか確認してきます. 良ければ次の画面へ進み，修正したければその画面を修正できます.
途中でセットアップを中断するには，CTRL+Cを押してください.

●「QBの概要と操作環境の説明」
このパッケージにはQuickBASICの概要と統合環境の操作方法を分かりやすく説明した，「QBチュートリアル」が含まれています. このメニューを選ぶと「QBチュートリアル」を起動します. 説明が終わるとMS-DOSに戻ります. 必要があれば，再度セットアッププログラムを起動してください.

【何かキーを押して下さい】

▲2枚目

QuickBASIC セットアップ ユーティリティ　　Version 1.00

■フロントプロセッサの組み込み　　(3/4)

QuickBASICに日本語入力システムを組み込む場合は，お使いのフロントプロセッサのマニュアルを参照して各自おこなってください．なお，ご使用になれるフロントプロセッサは，ATOK6，MS$KANJI API 仕様のもの（VJE－β Ver.2.0以上，E1など）です．
フロッピーディスクにセットアップした場合は，「QB起動ディスク」にフロントプロセッサのデバイスドライバを転送し，辞書は「QBワークディスク」に転送することをお勧めします．辞書のサイズに対して「QBワークディスク」のディスク容量が不足する場合は，このディスクに転送されているヘルプファイルとサンプルプログラムを削除してください．フロントプロセッサの組込みの方法は，ご利用のフロントプロセッサのマニュアルを参照してください．

■ドキュメントファイルについて

QuickBASICをお使いになる前に，ドキュメントファイルをお読みください．「セットアップディスク」には，README.DOC(機種依存情報，制限事項)，SAMPLE.DOC(サンプルの説明)，QBE.DOC(エミュレータ版QBの使い方)が含まれています．なお，「プログラムディスク」¥MOUSE ディレクトリには，MOUSE.DOC(マウスドライバの説明)が含まれています．必要に応じてご参照ください．

【何かキーを押して下さい】

▲3枚目

QuickBASIC セットアップ ユーティリティ　　Version 1.00

■セットアッププログラムで転送されないファイルについて　　(4/4)

このセットアッププログラムは，QuickBASIC をご利用いただく環境を作ることを目的としています．マスターディスクに含まれるすべてのファイルが転送されるわけではありません．転送されないファイルについては，PACKING.LST を参照してください．PACKING.LST は，「セットアップディスク」に含まれています．

■CONFIG.SYS，AUTOEXEC.BAT

このセットアッププログラムは，フロッピーディスクへセットアップしたとき「QB起動ディスク」に CONFIG.SYS ，AUTOEXEC.BAT を作成します．
ハードディスクの場合は指定されたディレクトリに，NEW-CONF.SYS，NEW-VARS.BAT を作成しますので，参考にして CONFIG.SYS，AUTOEXEC.BAT を修正してください．
なお，MS-DOS のバージョンによっては プリンタドライバ "PRINT.SYS" を組み込む必要があります．MS-DOS のマニュアルを参照して CONFIG.SYS を書き換えてください．

【何かキーを押して下さい】

▲4枚目

2.3 セットアップユーティリティメニュー

セットアッププログラムが起動して、次々にメッセージを読み、セットアップ画面を5回切り換えると、メニューが表示されます。あとは、このメニューに従って、

1. Quick BASICの「起動ディスク」の作成
2. QBの使い方
3. 終了

の中から「1.」を選択します。

```
QuickBASIC セットアップ ユーティリティ                Version 1.00

     ********* < メニュー > ********************************
       1. QuickBASICの利用環境を作成します。
       2. QBの概要、操作方法の説明を行います。
       3. セットアップを終了します。
     *******************************************************

      ●メニューを選択して下さい。・・・・・・・・・・・・?1

-----------------------------------------------------------------
  QuickBASICの利用環境を作成します。
  始めてお使いになる方は必ず 1のメニューを選んでください。
  途中でセットアップを中断するには、CTRL+CもしくはSTOPキーを押して
  ください。この場合は、再度セットアップを起動して最初から行う必要があります。

  2のメニューを選択すると、QuickBASICの利用環境の作成は行わずに、
  QBの概要や操作方法を説明する「QBチュートリアル」が起動します。
  利用環境の設定を行う場合は1を選択してください。
```

1 Quick BASICの「起動ディスク」作成

メニューから「1．Quick BASICの利用環境を作成します。」を選択します。

セットアッププログラムが、いくつかの質問をします。各質問についての説明が画面の下部に表示されるので、それぞれをよく読んで、質問に答えてください。理解できなかったところは、とりあえず「パス」の気持ちで、セットアップユーティリティが指示している番号のまま、先へ進めます。

1画面ごとに、その画面の設定でよいかどうか確認してきます。よければ次の画面へ進み、修正したければ、その画面を修正できます。終了すると、メニューに戻りますから、「3．終了」を選択します。

途中でセットアップを中断するには、[CTRL]+[C]か[STOP]キーを押します。

質問中の「転送元ドライブ」というのは、Quick BASICのマスターディスクを入れるディスクドライブのことです（たぶん、Aドライブになっていると思います）。

メニューの「2．QBの概要と操作環境の説明を行います。」には、Quick BASICの概要を説明した、「QBチュートリアル」が含まれています。このメニューを選ぶと「QBチュートリアル」を起動します。特にセットアップと関係ありませんので、画面の説明を読んで指示に従ってください。説明が終わるとMS-DOSに戻ります。もう一度、見直しをしたいときには、再度、SETUP［リターン］と入力して、セットアッププログラムを起動してください。

2 「起動ディスク」のファイルを見る

セットアップされた「起動ディスク」の中を見ます。A:ドライブに、たった今、セットアップされた「起動ディスク」を入れます。

```
DIR A:［リターン］
```

```
A>DIR

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    24931  88-07-13    0:00
BIN          <DIR>     90-05-02   11:16
LIB          <DIR>     90-05-02   11:17
AUTOEXEC BAT       73  90-05-02   11:24
CONFIG   SYS       51  90-05-02   11:24
        5 個のファイルがあります.
   131072 バイトが使用可能です.

A>

     C1    CU    CA    S1    SU     VOID   NWL   INS   REP   ^Z
```

さらに、CONFIG.SYSを見てみます。

TYPE CONFIG.SYS［リターン］

```
SHELL=A:¥COMMAND.COM A:¥ /P
FILES=20
BUFFERS=10
```

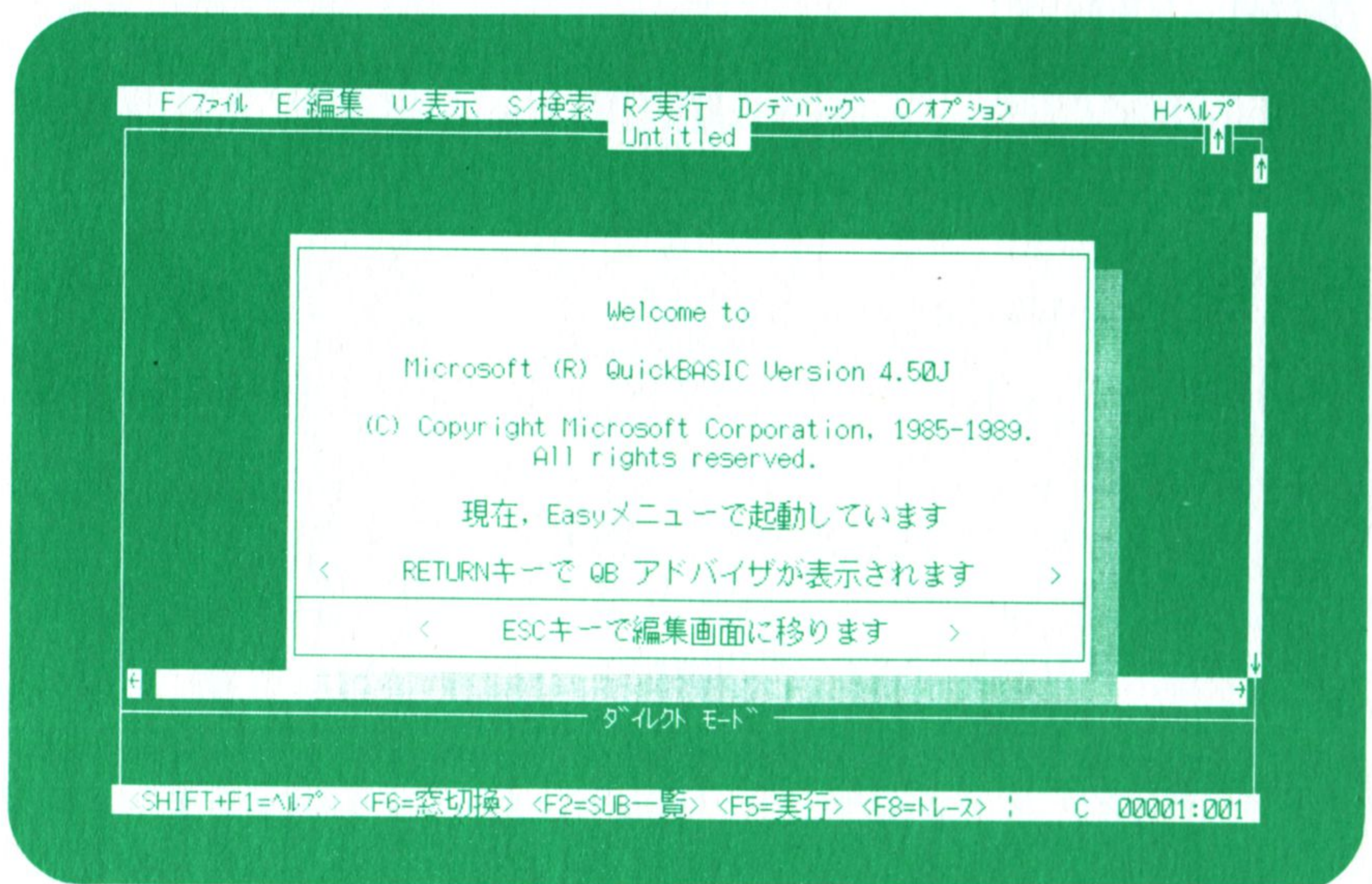
続いて、AUTOEXEC.BATの中も見ます。

TYPE AUTOEXEC.BAT［リターン］

```
PATH=A:¥BIN
SET LIB=A:¥LIB
SET INCLUDE=B:¥INCLUDE
SET TMP=B:¥
MOUSE
```

CONFIG.SYSのファイルは、日本語エンドフロントプロセッサの組み込みがされていません。また、AUTOEXEC.BATは、自動的にQuick BASICが起動できるようにはなっていないのがわかると思います。もし、日本語の入力を考えないのであれば、このまま使うことができます。

1．Aドライブに「QB起動ディスク」を入れる
2．Bドライブにワークディスクを入れる
3．リセットボタンを押す
4．画面にA>が表示されたのを確認する
5．QB［リターン］と入力する

```
F/ﾌｧｲﾙ  E/編集  V/表示  S/検索  R/実行  D/ﾃﾞﾊﾞｯｸﾞ  O/ｵﾌﾟｼｮﾝ                H/ﾍﾙﾌﾟ
                              Untitled

                       Welcome to

          Microsoft (R) QuickBASIC Version 4.50J

      (C) Copyright Microsoft Corporation, 1985-1989.
                   All rights reserved.

              現在, Easyメニューで起動しています

     <   RETURNキーで QB アドバイザが表示されます   >

          <    ESCキーで編集画面に移ります    >

                       ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ
<SHIFT+F1=ﾍﾙﾌﾟ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=ﾄﾚｰｽ> |    C  00001:001
```

パソコンの電源を切る

MS-DOSを使っているアプリケーションソフトを終了させても画面上にA>が表示されます。この状態になったらパソコンの電源を切ることができます。

A>

2.4 日本語フロントエンドプロセッサの組み込み

Quick BASICに日本語入力システムを組み込む場合は、好みのフロントプロセッサを組み込むことになります。ここでは、Quick BASICが推奨しているATOK6を組み込むこととします。そのほかのフロントプロセッサの場合は、それぞれのマニュアルを参照して各自で行なってください。なお、フロントプロセッサは、次のような制限が設けられています。

ATOK6、MS$KANJI API仕様のもの(VJE-β V2.0以上、E1など)です。

フロントプロセッサのセットアップは、フロントプロセッサのデバイスドライバを「QB起動ディスク」に転送し、辞書は「QBワークディスク」に転送することとします。「QBワークディスク」のディスク容量が不足する場合は、このディスクに転送されているヘルプファイルとサンプルプログラムを削除してください。

組み込みは、次の順番で行ないます。

1．「一太郎V3」を起動／終了
2．フロントプロセッサのデバイスドライバ（ATOK6A.SYS、ATOK6B.SYS）を転送
3．辞書（ATOK.DIC）を転送
4．Aドライブに「一太郎V3のユーティリティディスク」を入れる
5．ATUT［リターン］
6．AUTOEXEC.BATの書き換え
7．終了

1 デバイスドライバの転送

まず、「一太郎V3」を起動して終了させ、A>が表示されている状態で、Bドライブに「QB起動ディスク」を入れて、ATOK6A,Bを転送します。

```
A>COPY ATOK6?.* B:
ATOK6A.SYS
ATOK6B.SYS
        2 個のファイルをコピーしました.

A>
```

2 辞書の転送

Bドライブに「QBワークディスク」を入れます。Aドライブは、「一太郎V3」のシステムディスクのままです。

```
A>COPY A:ATOK.DIC B:
        1 個のファイルをコピーしました.

A>
```

もし、「QBワークディスク」の容量不足で転送がうまくいかなかった場合には、「6.辞書組み込みで削除できるファイル」を参照してください。

3 デバイスドライバの組み込み

Aドライブの「一太郎V3」のシステムディスクを一太郎のユーティリティディスクと入れ替えます。Bドライブへは、「QB起動ディスク」を入れます。ここでは、「QB起動ディスク」のCONFIG.SYSを一太郎のユーティリティディスクを使って、書き換えようというわけです。Aドライブに一太郎のユーティリティディスクを入れたら、ATUT［リターン］と入力します。ATOK6環境登録ユーティリティが起動します。

辞書ドライブは、カーソル移動キーでBドライブを指定します。また、入力位置は「エコー」を選択します。「エコー」はQuick BASICが起動したときに、カーソルのある位置に日本語が入力され、「システムライン」は入力された日本語がいったん画面の下に表示されたあと、カーソル位置に移動します。

```
***   ATOK6 環境登録ユーティリティ   ***
                                ATOK単体用

辞書ドライブ         [ カレント   A  B  C  D ]
辞書ファイル名       [ ATOK.DIC     ]
外字ドライブ         [ カレント   A  B  C  D ]
外字ファイル名       [ GAIJ         ]
モード選択キー       [ f・10   ROLLUP ]
入力モード           [ ローマ字漢字  カナ漢字 ]
句読点モード         [ ，．「」／  、。「」・ ]
辞書学習             [ する  しない ]
入力位置             [ システムライン  エコー ]
変換モード           [ 連文節   単文節 ]
コード体系           [ JIS   シフトJIS   区点 ]

移動：↑↓キー，←→キー   確定：改行キー   中止：ESCキー
 C1   CU   CA   S1   SU    VOID  NWL  INS  REP  ^Z
```

CONFIG.SYSの更新を聞いてきますので、［はい］を選択してください。更新を行なうドライブはそのあとで聞いてきます。更新のドライブは「B」です。CONFIG.SYSの書き換えは、これで終了ですから楽でしょう。

書き換えたCONFIG.SYSを見てみます。

```
B:¥>TYPE CONFIG.SYS
SHELL=A:¥COMMAND.COM A:¥ /P
FILES=20
BUFFERS=10
DEVICE=ATOK6A.SYS /N=ATOK.DIC /L=GAIJ /R=0 /M=0 /T=1 /S=1 /E=1 /B=0 /C=0
DEVICE=ATOK6B.SYS

B:¥>

 C1   CU   CA   S1   SU    VOID  NWL  INS  REP  ^Z
```

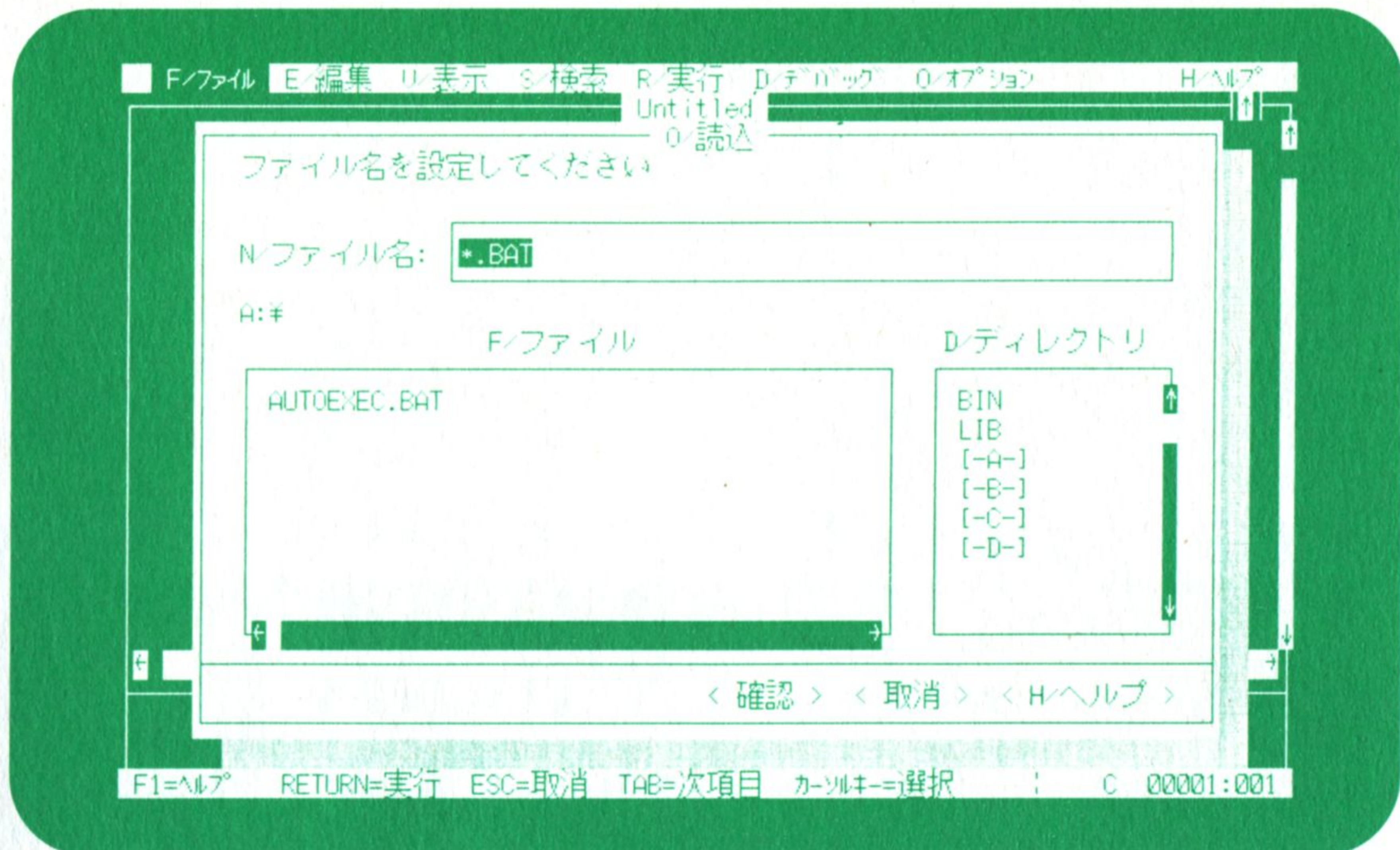
4　AUTOEXEC.BATの書き換え

自動的にQuick BASICが起動するようにAUTOEXEC.BATを書き換えます。現在、

```
PATH=A:¥BIN
SET LIB=A:¥LIB
SET INCLUDE=B:¥INCLUDE
MOUSE
```

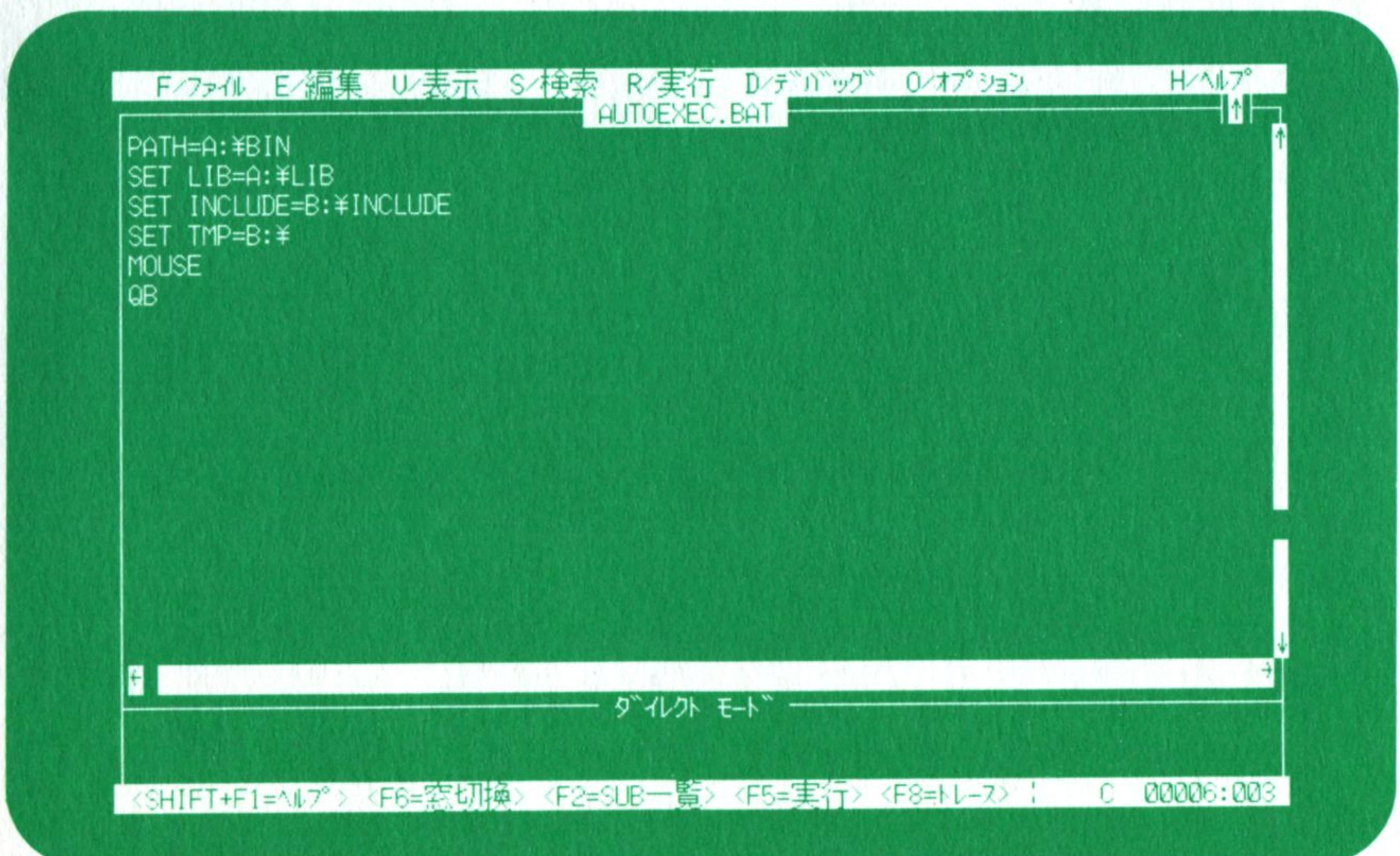
次のようになっているファイルにQBの2文字を追加するだけです。

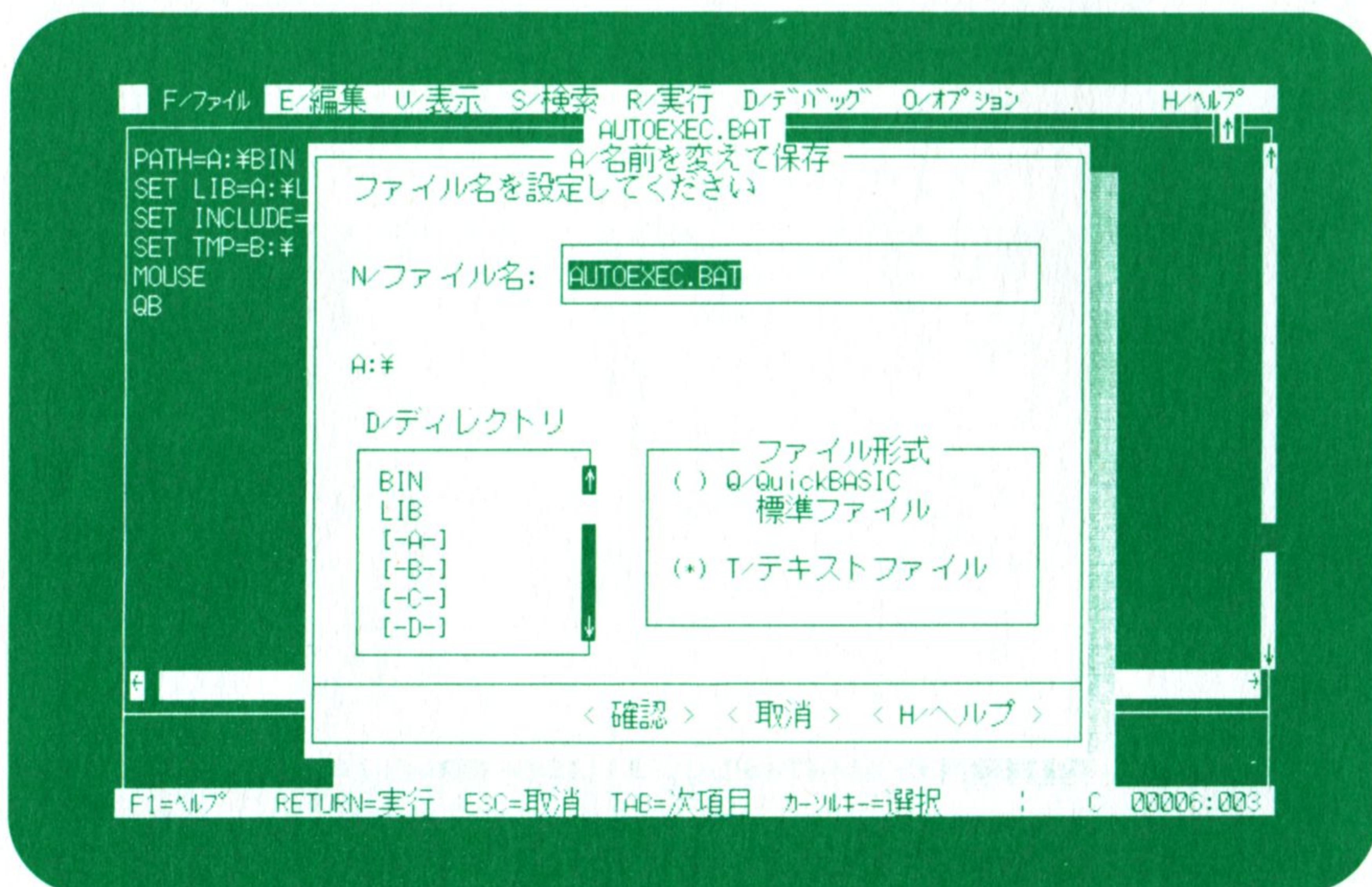
1．「QB起動ディスク」をAドライブに入れて、リセットボタンを押します。
2．A>になっているのを確認して、QB［リターン］と入力します。
3．Quick BASICが起動しますので、[ESC]キーを押します。
4．[GRPH]キーを押して、［リターン］キーを押すと、サブメニューが表示されます。

5．「O/読込み」をカーソル移動キーで選択します。
6．［リターン］キーを押すと「O/読み込み」画面が表示されますので、*. BAT［リターン］と入力します。

7．[TAB]キーを押します。

8．小さなカーソルが見えています。[リターン]キーを押して、AUTOEXEC.BATのファイル名を反転させます。

9．[リターン]キーを押します。

10．AUTOEXEC.BATの内容が表示されますから、MOUSEのあとにQBを入力します。

11. [GRPH]キーを押して、「F/ファイル」のメニューを出します。操作4と同じです。今度は、「A/名前を変えて保存」を選択します。
12. 「保存」メニューが表示されたら、ファイル名の書き換えをやらずにそのまま、［リターン］キーを押します。
13. 再度、[GRPH]キーを押して、「F/ファイル」メニューをひらいたあと、「X/終了」を選択して、Quick BASICを終了させます。
14. リセットボタンを押すと、今度は、自動的にQuick BASICが起動します。

```
F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション   H/ヘルプ
                        AUTOEXEC.BAT
PATH=A:¥BIN          A/名前を変えて保存
SET LIB=A:¥L   ファイル名を設定してください
SET INCLUDE=
SET TMP=B:¥
MOUSE          N/ファイル名: AUTOEXEC.BAT
QB
               A:¥

               D/ディレクトリ
                BIN                  ファイル形式
                LIB              ( ) Q/QuickBASIC
                [-A-]                標準ファイル
                [-B-]
                [-C-]            (*) T/テキストファイル
                [-D-]

                     < 確認 >  < 取消 >  < H/ヘルプ >
F1=ヘルプ  RETURN=実行  ESC=取消  TAB=次項目  カーソルキー=選択    :   C  00006:003
```

5　困ったもんだMS-DOS

日本語フロントエンドプロセッサのデバイスドライバの登録して、自動的に起動できるようにセットアップできても、MS-DOSのバージョンによっては、プリンタが動かないことがあります。プリンタを利用するためには、MS-DOSのシステムディスクに付属しているプリンタドライバをCONFIG.SYSに組み込む必要があります。使用するMS-DOSのバージョンを確認してください。

さらに始末の悪いことに、ドライバはバージョンによってはPRINT.SYSであった

り、PRINT.EXEであったりということで、「気をつけて……」としかいいようがありません。

1．PRINT.SYSを必要としいるバージョンのMS-DOSを使っている人は、AドライブにMS-DOSシステム、Bドライブに「QB起動ディスク」を入れます。
2．COPY A:PR*.* B:［リターン］
このとき、拡張子が.SYSになっているか、.EXEになっているかを確認しておきます。それによって、CONFIG.SYSの内容が少し変わります。
3．CONFIG.SYS の例（2例掲載しておきます）

```
FILES = 20
BUFFERS = 20
日本語フロントエンドプロセッサ
DEVICE = PRINT.SYS
```

```
FILES = 20
BUFFERS = 20
日本語フロントエンドプロセッサ
DEVICE = PRINT.EXE
```

MS-DOSのバージョン		PRINT.SYS
MS-DOS Ver.2.0	(PS98-121-XXX)	不要
	(PS98-122-XXX)	不要
	(PS98-123-XXX)	不要
MS-DOS Ver.3.1	(PS98-125-XXX)	不要
	(PS98-127-XXX)	不要
	(PS98-129-XXX)	不要
	(PS98-011-XXX)	必要
MS-DOS Ver.3.3	(PS98-013-XXX)	必要
MS-DOS Ver.3.3A	(PS98-015-XXX)	必要

6　辞書組み込みで削除できるファイル

日本語フロントプロセッサの辞書を組み込むのは、「QBワークディスク」です。このワークディスクには、次のようなファイルが入っています。ディレクトリの頭に◎印のついたファイルは削除することができます（×印はダメ）。

```
◎¥ADVR  /DIR/________QB45ENER.HLP
                    |__QB45QCK.HLP
                    |__QB45USER.HLP
◎¥DOC  /DIR/_________README.DOC
                    |__SAMPLE.DOC
×¥INCLUDE  /DIR/_____GEN.BI
                    |__GNCONST.BI
                    |__GRAPH.BI
                    |__HWCONST.BI
                    |__MOUSE.BI
                    |__QB.BI
◎¥SOURCE  /DIR/______¥ADVR_EX  /DIR/____*.BASが20ファイル
                    |__¥BOOK  /DIR/_______掲載例プログラム、他、37ファイル
                    |__¥EXAMPLE  /DIR/____19ファイル
                    |__DEMO1.BAS
                    |__DEMO2.BAS
                    |__DEMO3.BAS
                    |__QCARDS.BAS
                    |__QCARDS.DAT
                    |__REMLINE.BAS
                    |__SORTDEMO.BAS
                    |__TORUS.BAS
                    |__WALTZ.BAS
```

実際のファイルの削除例：¥SOURCEを削除してみます。。

```
1．DEL ¥SOURCE              2．DEL ¥SOURCE¥ADVER_EX
3．DEL ¥SOURCE¥BOOK         4．DEL ¥SOURCE¥EXAMPLE
5．RD ¥SOURCE¥ADVER_EX      6．RD ¥SOURCE¥BOOK
7．RD ¥SOURCE¥EXAMPLE       8．RD ¥SOURCE
```

2.5 ハードディスクへのセットアップ

ハードディスクのセットアップを行ないます。このときの状態は、Aドライブはハードディスク。B、Cがフロッピーディスクドライブとします。

1　セットアップメニューの選択

Bドライブにマスター「セットアップディスク」を入れて、A>の状態をB>に変更させます。

```
A>B:［リターン］
B>SETUP［リターン］
```

メッセージ画面を4枚めくると、セットアップメニューになります。

2　利用環境の選択

メニュー画面から、「1．Quick BASICの利用環境を作成します。」を選択します。ここまでは、フロッピーディスクのセットアップと同じです。

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00

        ********* ＜メニュー＞ **********************************
          1. QuickBASICの利用環境を作成します。
          2. QBの概要、操作方法の説明を行います。
          3. セットアップを終了します。
        ********************************************************

          ●メニューを選択して下さい。・・・・・・・・・・・？1

------------------------------------------------------------------
 QuickBASICの利用環境を作成します。
 始めてお使いになる方は必ず 1のメニューを選んでください。
 途中でセットアップを中断するには、CTRL+CもしくはSTOPキーを押して
 ください。この場合は、再度セットアップを起動して最初から行う必要があります。

 2のメニューを選択すると、QuickBASICの利用環境の作成は行わずに、
 QBの概要や操作方法を説明する「QBチュートリアル」が起動します。
 利用環境の設定を行う場合は1を選択してください。
```

転送先のドライブは、「セットアップディスク」の入っているドライブです。この場合は、Bを指定します。

続いて、インストールドライブの指定です。こんどは、ハードディスクを指定します。Aを入力してください。

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00
 ●転送元のドライブ名を設定してください？B
 ●どのドライブにインストールしますか？A

------------------------------------------------------------------
 QuickBASICをインストールするドライブ名を入力してください。
 フロッピーディスクにインストールする場合は、以下の準備が整っていることを確
 認してください。
   1. フォーマット済みフロッピーディスクの準備。
      (DD版の時は4枚、HDの時は3枚のディスクが必要です。)
   2. パッケージに添付されている黒色の作業ディスクラベルを貼る。
   3. [QB起動ディスク]にMS-DOSを転送する。
 準備ができていない時は、CTRL+Cでセットアップを終了し、ディスクを用意
 してから再度セットアップを起動してください。
```

フロッピーディスクのインストールと違うのは、セットアップメニューの中にパスディレクトリの設定が入ってきます。テンポラリファイルは、RAMディスクを使っていたら、D:¥にしておくと、ハードディスクのメモリーの節約ができます。

```
QuickBASIC セットアップ ユーティリティ                    Version 1.00

 ●QuickBASIC利用時のパスディレクトリの設定 <挿入>
 +-------------------+-----------------------------------+
 | 実行 ﾌｧｲﾙ          |A:¥QB45¥BIN                        |
 | ｲﾝｸﾙｰﾄﾞ ﾌｧｲﾙ       |A:¥QB45¥INCLUDE                    |
 | ﾗｲﾌﾞﾗﾘ ﾌｧｲﾙ        |A:¥QB45¥LIB                        |
 | HELP ﾌｧｲﾙ          |A:¥QB45¥ADVR                       |
 | ｿｰｽ ﾌｧｲﾙ           |B:¥QB45¥SOURCE                     |
 |>ﾃﾝﾎﾟﾗﾘ ﾌｧｲﾙ        |D:¥                                |
 +-------------------+-----------------------------------+
--------------------------------------------------------------
 テンポラリファイルを作成するドライブ、ディレクトリを設定します。テンポラリ
 ファイルは、LINK.EXE が利用します。RAMディスク、もしくはハードディスクが
 ある場合は、そのドライブを設定してください。
 《キーの操作方法》
 [ESC]キー     ：全ての設定を確定します
 [CR]キー      ：全ての設定を確定します
 [←][→]キー ：カーソルを左右に移動します
 [↑][↓]キー ：カーソルを上下に移動します
 [INS]キー     ：現在のモード（挿入モードと上書きモード）を変更します
 [BS] キー     ：カーソル位置の左側の文字を消し、カーソルを左に移動します
 [DEL]キー     ：カーソル位置の文字を消します
```

あとは、画面上にマスターディスクの差し替えの指示が出ます。それに従っていけば、自動的にファイルのコピーを行なってくれます。コピーが一段落するとメニューに戻ります。「3」を選択してセットアップは終了です。

3 CONFIG.SYSとAUTOEXEC.BATの書き換え

自動的にQB45というディレクトリが作成され、ディレクトリの中に次のようなファイルが入っています。

```
CD¥QB45 [リターン]
DIR [リターン]
```

```
A:¥>CD¥QB45

A:¥QB45>DIR

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥QB45

.              <DIR>      90-05-02   12:34
..             <DIR>      90-05-02   12:34
BIN            <DIR>      90-05-02   12:34
LIB            <DIR>      90-05-02   12:35
INCLUDE        <DIR>      90-05-02   12:35
SOURCE         <DIR>      90-05-02   12:35
ADVR           <DIR>      90-05-02   12:36
DOC            <DIR>      90-05-02   12:37
NEW-VARS BAT         88   90-05-02   12:40
NEW-CONF SYS         51   90-05-02   12:40
       10 個のファイルがあります.
 28794880 バイトが使用可能です.

A:¥QB45>

 C1  CU  CA  S1  SU   VOID  NWL  INS  REP  メニュー
```

画面上のNEW-VARS.BATはQuick BASIC　V4.5用のAUTOEXEC.BATです。

また、NEW-CONF.SYSは、CONFIG.SYSの参考ファイルですから、ハードディスクのAUTOEXEC.BATとCONFIG.SYSの書き換えの参考にしてください。

▼NEW-VARS.BATの内容

```
PATH=A:¥QB45¥BIN
SET LIB=A:¥QB45¥LIB
SET INCLUDE=A:¥QB45¥INCLUDE
SET TMP=D:¥
```

▼NEW-CONF.SYSの内容

```
SHELL=A:¥COMMAND.COM A:¥ /P
FILES=20
BUFFERS=10
```

2.6 Quick BASICの起動と終了

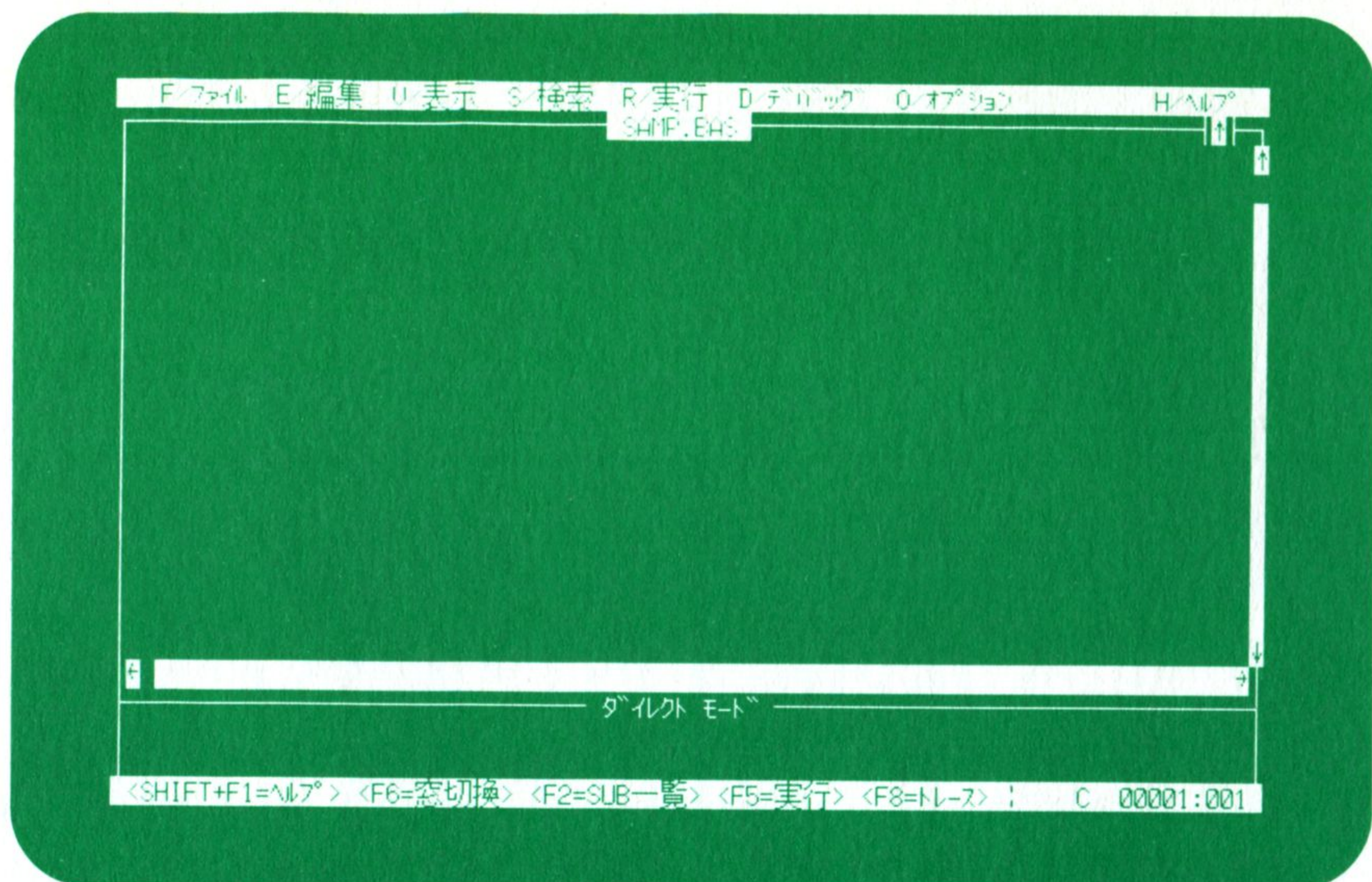
セットアップが無事終了しました。いよいよ、Quick BASICの起動と操作と終了の一連の流れを経験します。まず、Aドライブに「QB起動ディスク」を、Bドライブに「ワークディスク」を入れます。「ワークディスク」は、単にMS-DOSでフォーマットされただけのディスクで、ここに作成したプログラムを保存します。

1 Quick BASICの起動

「QB起動ディスク」をAドライブに入れたあと、リセットスイッチを押します。画面上に、MS-DOSのバージョン表示、ATOKのシステムとコピーライトの表示後、自動的にQuick BASICが起動します。自動的に起動している関係上、画面の中央のプログラムファイル名のところに、「UNTITLE」（ファイル名ナシ）と表示されます。

この状態が気持ちの悪い人は、「QB起動ディスク」のAUTOEXEC.BATのQBを削除して、Quick BASICを起動させます。画面上にA>が表示されたら、QB B:プログラム名と入力して、Quick BASICを起動させると、Bドライブにファイル名.BASという名前でファイルが作られます。

例：A>QB B:SAMP.BAS [リターン]

F/ﾌｧｲﾙ E/編集 V/表示 S/検索 R/実行 D/ﾃﾞﾊﾞｯｸﾞ O/ｵﾌﾟｼｮﾝ H/ﾍﾙﾌﾟ
SAMP.BAS
ﾀﾞｲﾚｸﾄ ﾓｰﾄﾞ
<SHIFT+F1=ﾍﾙﾌﾟ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=ﾄﾚｰｽ> ¦ C 00001:001

メインメニューバーに表示されている8項目の頭文字を入力するか、カーソル移動キー［←］［→］を使って、項目を反転させたあと［リターン］キーを押します。反転されたメニューの項目によっては、さらに、サブのメニューがプルダウンで表示されます。この項目の選択も頭文字がカーソル移動キーで行ないます。

2 メニュー操作で使うキー

メニュー操作は、大きく分けると3つの方法があります。また、この方法は混在して使うことができます。

1．キーボードからの入力

[GRPH]キー…………押すことで、メニュー選択モードになります。
[←]、[→]……………メインメニューのカーソルを移動します。
[↑]、[↓]……………サブメニューカーソルを移動します。
[TAB]キー………… サブ（プルダウン）メニュー内の項目を次に進めます。
[SIFT]+[TAB]キー… サブ（プルダウン）メニュー内の項目を1つ前に戻します。
[リターン]キー…… メニューカーソルのある項目を実行します。
[ESC]……………… メニュー処理をキャンセルします。

２．マウス操作で入力

マウスカーソルをメニューの項目に移動して、左ボタンをクリック（押す）して選択します。

３．ショートカットキー操作

サブメニューの項目の右横に、示されているキーがショートカットキーです。このショートカットキーを使えば、メニューを表示して項目を選択する手間が省けます。

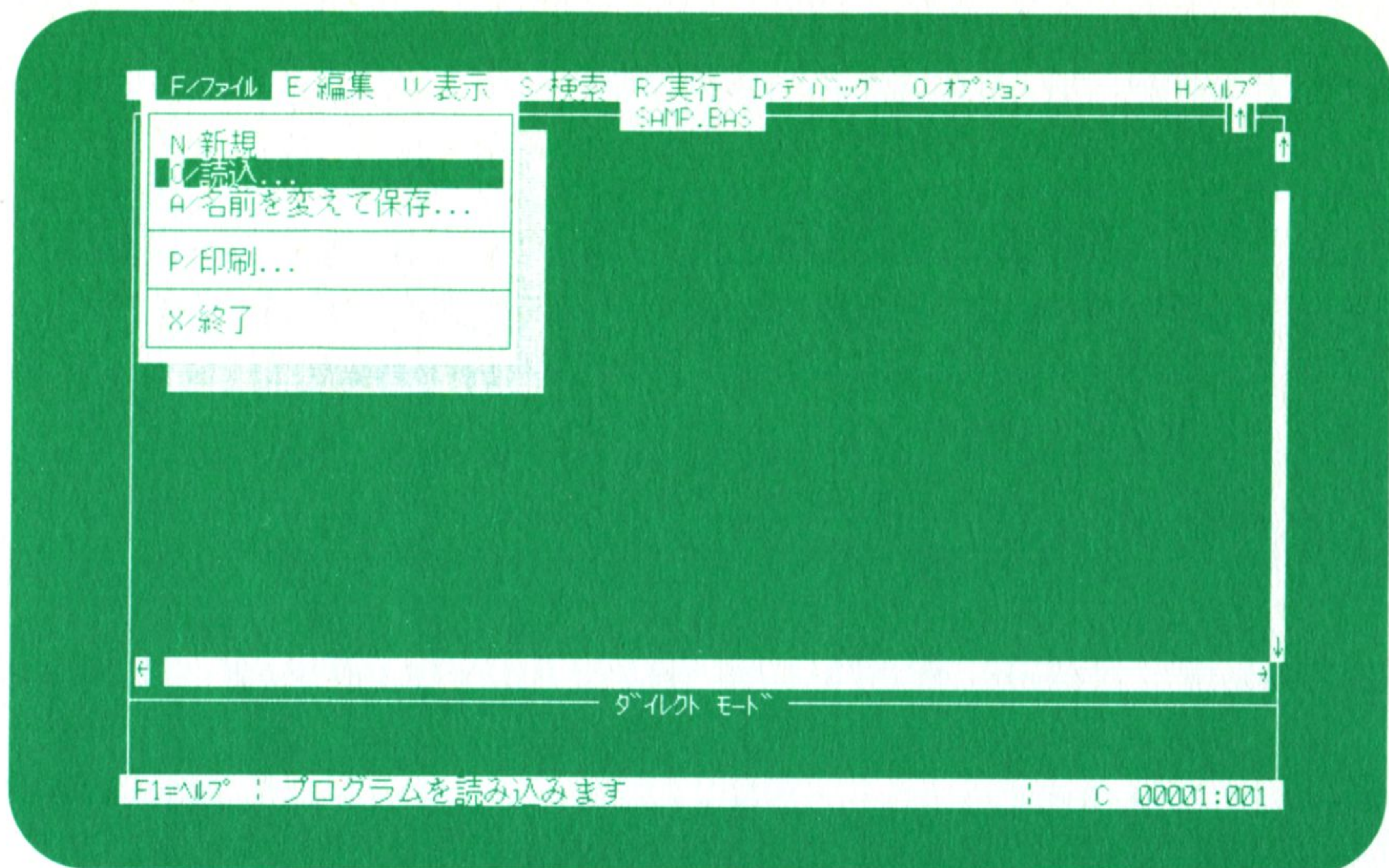
たとえば、プログラムの実行は、１のメニュー操作を行なうと、[GRPH]キーを押して、メインメニューバーにカーソルを移動させたあと、R/実行メニューからプルダウンメニューをひらいて、S/スタートコマンドを選択することになります。２のマウス操作においても、同じように、それぞれの項目をクリックしていきます。それが、ショートカットキーであれば、[SIFT]キーを押したまま、[f･5]を押すだけで、ただちに実行できます。

あわてて、まるごとショートカットキーを覚える必要はありません。また、いっきに覚えることは無理ですし、精神衛生上もよくありません。ここでは、2つのキー操作について覚えてください。

メインメニューバーへのカーソル移動は、[GRPH]キー
BASICの実行は、[SIFT]＋[f･5]キー

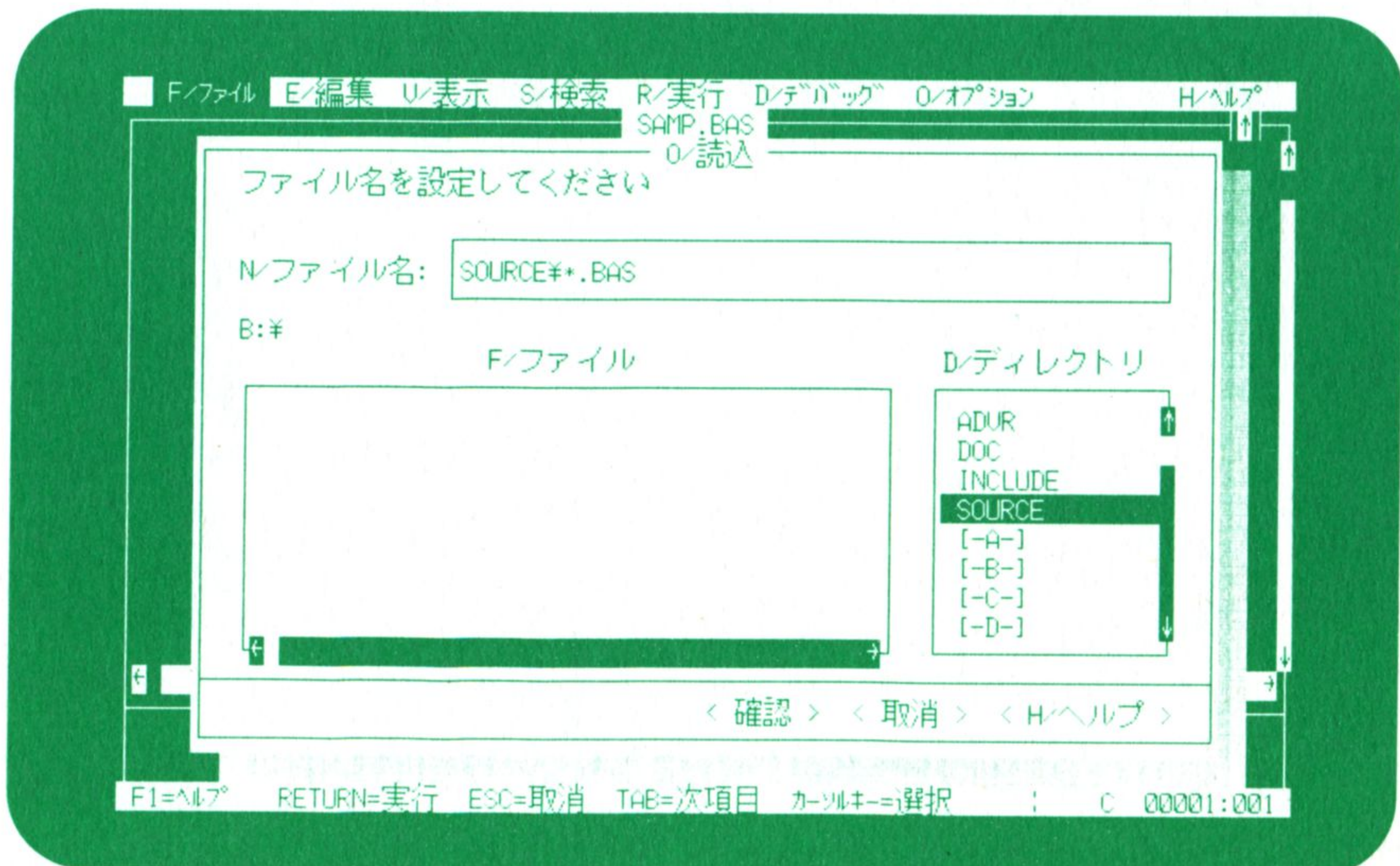
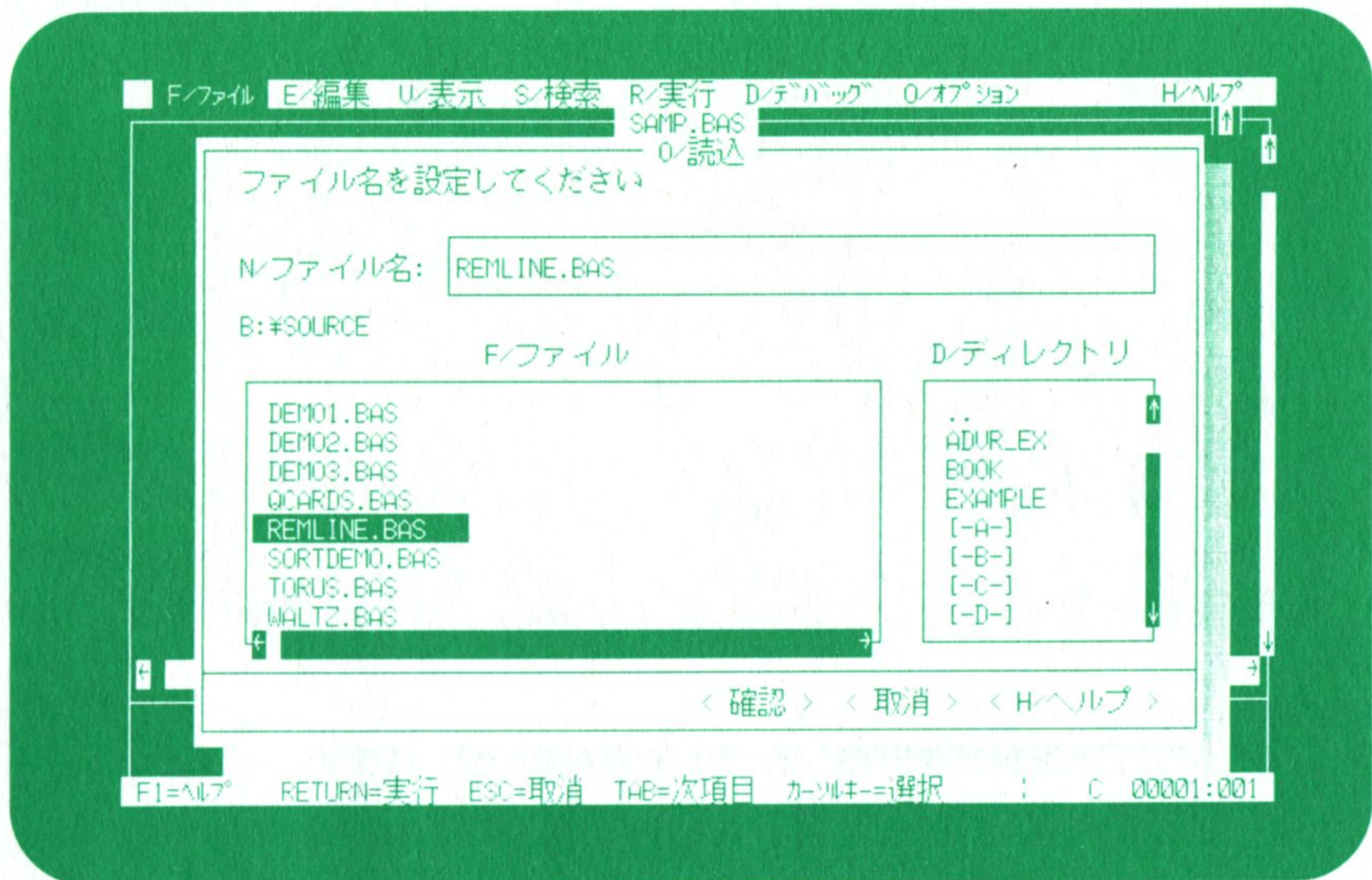
3　サンプルプログラムの実行

サンプルプログラムを画面に読み込んで、プログラムの実行を行ないます。

１．[GRPH]キーを押して、メインメニューバーにカーソルを移動させる。
２．F/ファイルの項目を反転させ、[リターン]キーを押す。

3．ファイル名の入力画面が表示される。

4．B:を入力して、ドライブの指定をBドライブにする（このとき表示されているファイル名を[BS]キーを使って消す必要はない）。

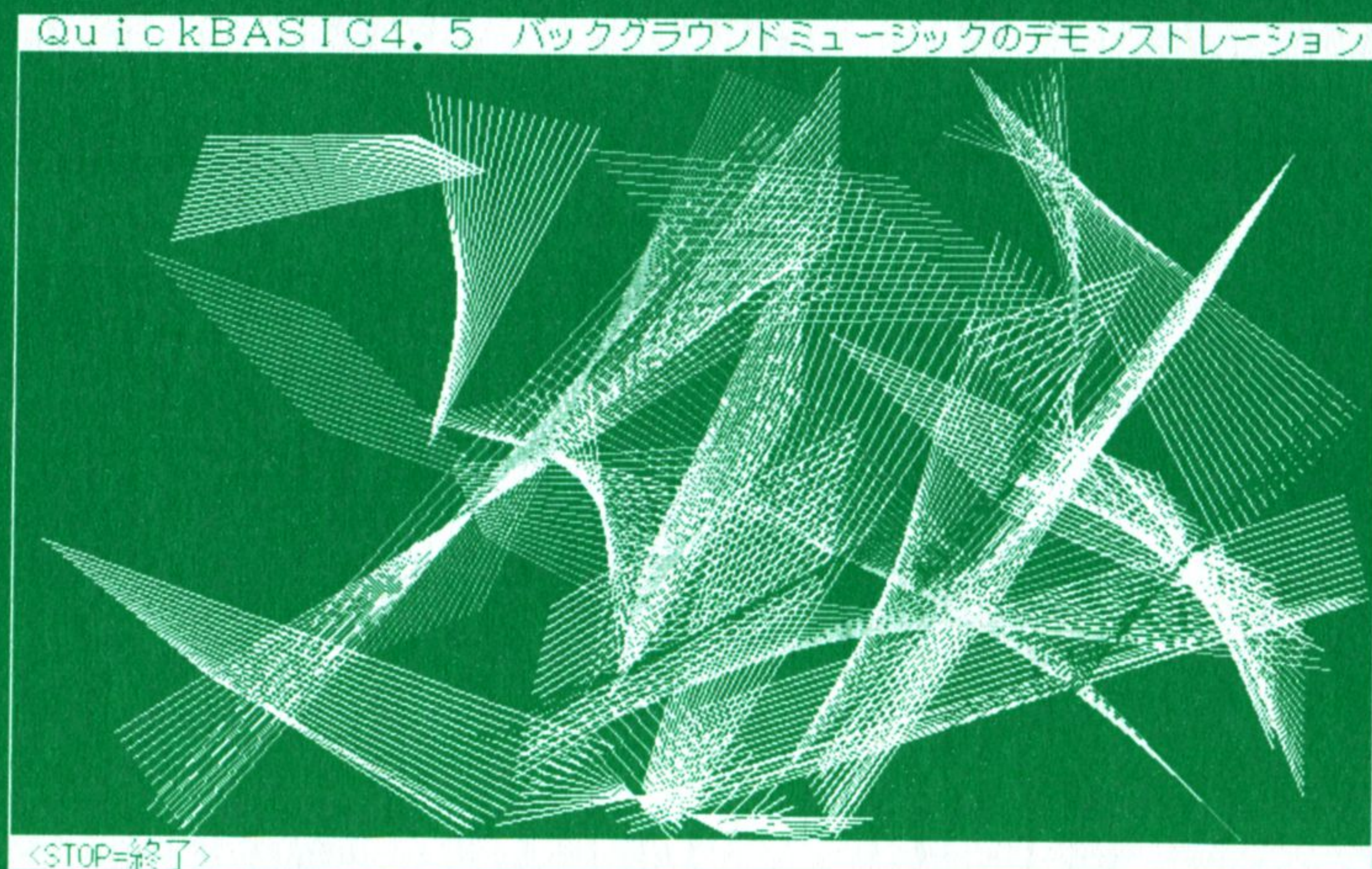
5．[TAB]キーを押して、ファイル名一覧のウィンドウの「D/ディレクトリ」へカーソルを移動させる。

6．サブディレクトリの「SOURCE」を選んで、[リターン]キーを押す。

7．拡張子「.BAS」のついたサンプルプログラムが表示されるので、[TAB]キーで、カーソルをジャンプさせたあと、カーソル移動キー（[←]　[→]　[↑]　[↓]）を使って、ファイルを選択する。

8．ここでは、一例として、WALTZ.BASを選択。[リターン]キーを押すと、画面にサンプルプログラムが読み込まれる。

9．BASICのプログラム実行のショートカットキーは、[SIFT]＋[f･5]でした。これで、プログラム表示の編集画面が切り換わって、WATZ.BASが実行される。画面表示をしながらバックグラウンドミュージックが流れる。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                              WALTZ.BAS
' このプログラム例は，バックグラウンドミュージックを演奏しながら他の処理を
' 行うための参考です．
'
' 注意: PC-9801E/F/M では，PLAY，SOUND で音階を出すことができないため BEEP
' 音しかなりません．'-----で囲まれた部分を削除してください．

'
CONST HighY = 383
CONST HighX = 639
CONST LowX = 0
CONST LowY = 16
WIDTH 80, 25
INPUT "線の本数を入力して下さい."; Num%
INPUT "影の数を入力して下さい."; Shad%
SCREEN 0
DIM X%(Num%, 1), Y%(Num%, 1)
DIM dX%(Num%, 1), dY%(Num%, 1), px%(Num%, 1, Shad%), pY%(Num%, 1, Shad%)
RANDOMIZE TIMER
                              ダイレクト モード
<SHIFT+F1=ヘルプ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=トレース> ¦    C  00001:001
```

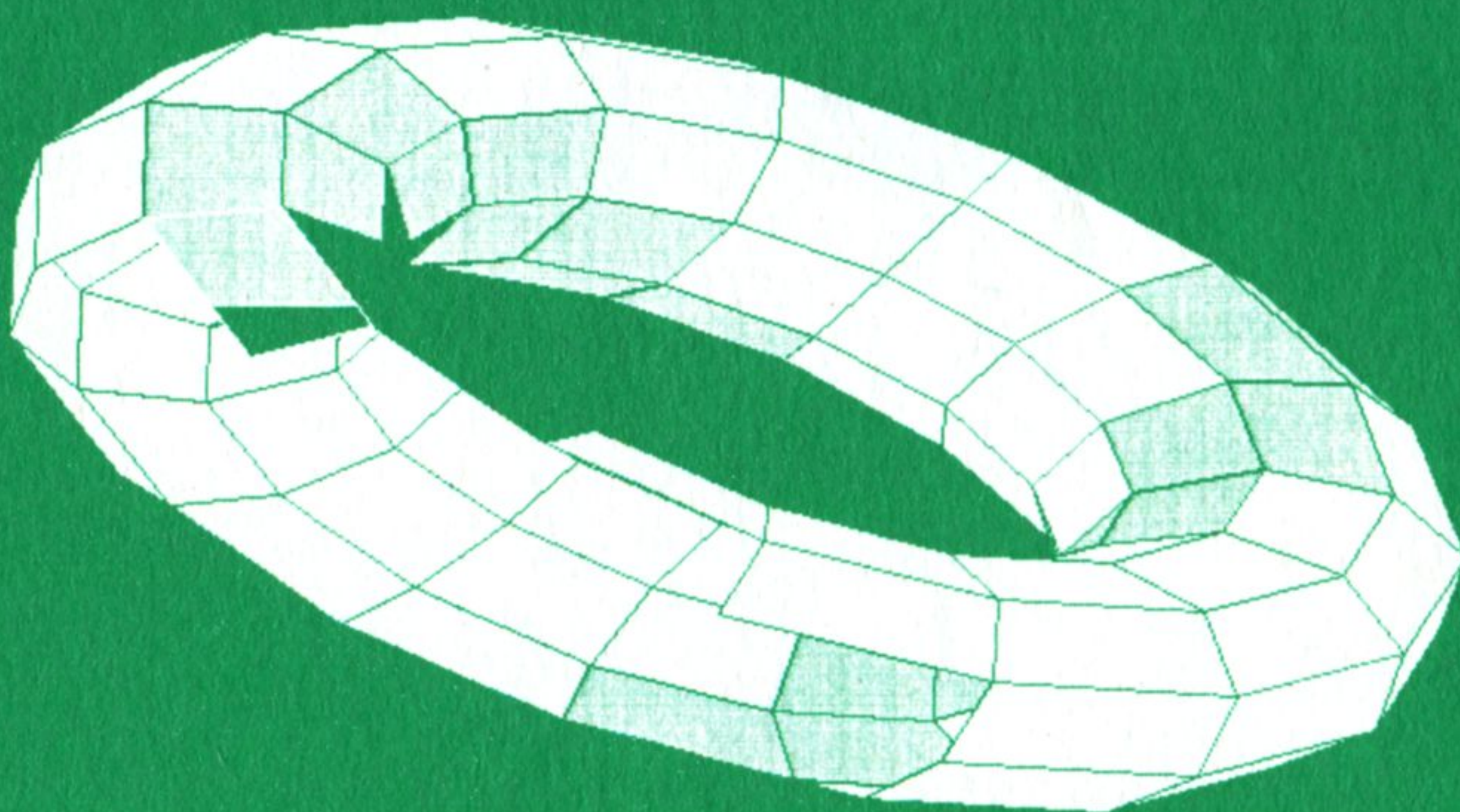
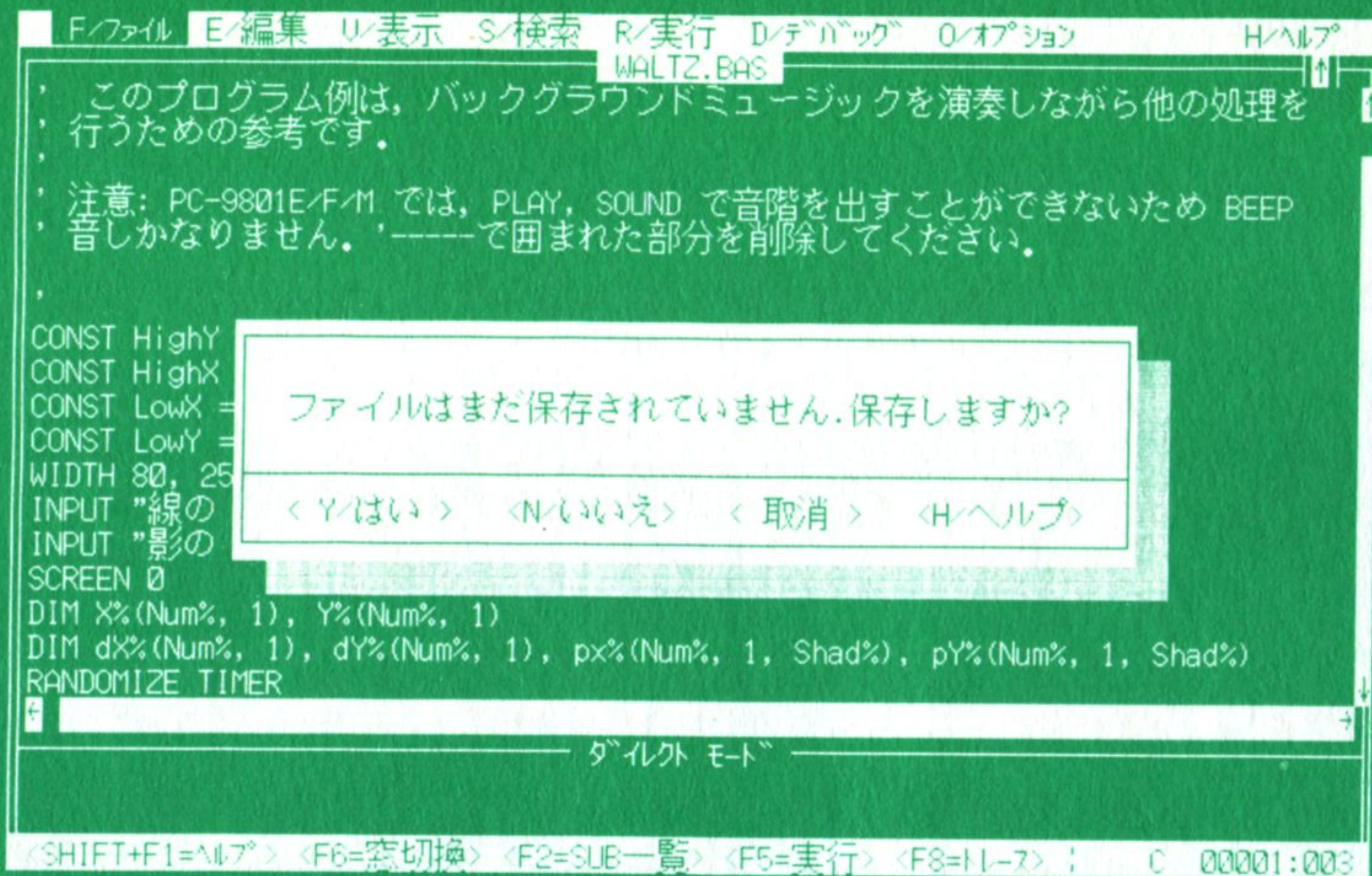
いろいろな、サンプルプログラムを走らせてみてください。サンプルプログラムには、プログラムを組むためのノウハウがたくさん詰まっています。

```
QuickBASIC4.5 バックグラウンドミュージックのデモンストレーション

<STOP=終了>
```

4 Quick BASICの終了

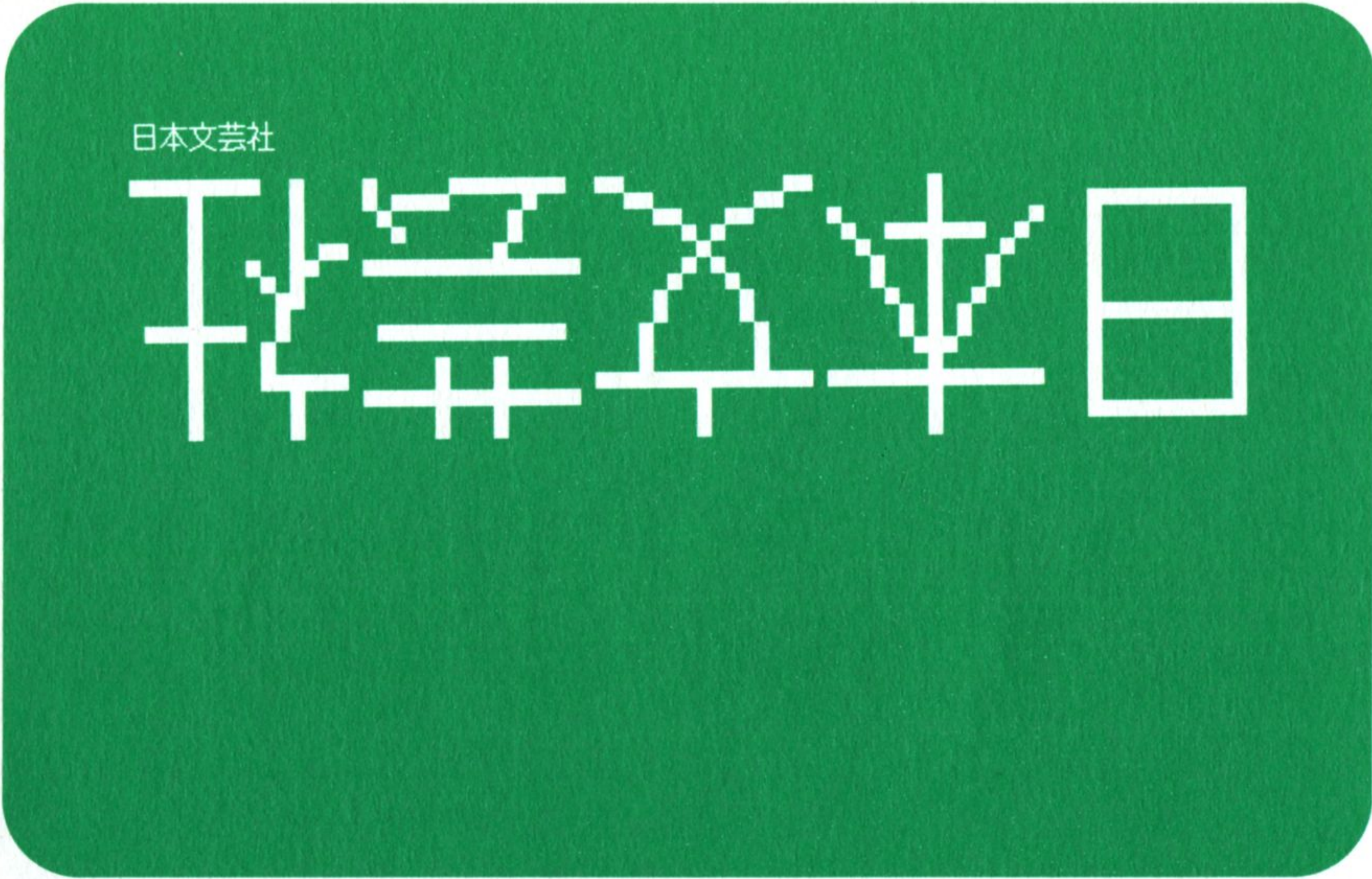
Quick BASICの終了もF/ファイルでプルダウンメニューを表示して、［↑］［↓］キーで、X/ファイルを選択します。メインメニューバーへのカーソルの移動は[GRPH]キーです。

```
F/ファイル E/編集 V/表示 S/検索 R/実行 D/デバッグ O/オプション H/ヘルプ
                        WALTZ.BAS
N 新規                 グラウンドミュージックを演奏しながら他の処理を
O 読込...
A 名前を変えて保存...
                       LAY, SOUND で音階を出すことができないため BEEP
P 印刷...              囲まれた部分を削除してください。

X 終了

CO
CONST LowX = 0
CONST LowY = 16
WIDTH 80, 25
INPUT "線の本数を入力して下さい."; Num%
INPUT "影の数を入力して下さい."; Shad%
SCREEN 0
DIM X%(Num%, 1), Y%(Num%, 1)
DIM dX%(Num%, 1), dY%(Num%, 1), px%(Num%, 1, Shad%), pY%(Num%, 1, Shad%)
RANDOMIZE TIMER
                        ダイレクト モード

F1=ヘルプ | QuickBASIC を終了します          |   C  00001:003
```

サンプルプログラムを書き換えた場合には、次のようなメッセージが表示され、プログラムの保存の有無を聞いてきます。今回は、サンプルプログラムを動かしてみるということであったので保存は行ないません。[N]キーを押します。

ついでというのも変ですが、サンプルプログラムの実行画面をあと２つ紹介しておきます。

▲TOROS.BASの実行画面

▼FONTMAG.BASの実行画面

日本文芸社

＜試してみよう＞

MS-DOSのCOPYコマンドの応用

MS-DOSという名前は、マイクロソフト社のディスクドライブオペレーティングシステムという意味です。MS-DOSのコマンドでOSを体感してみましょう。ドライブBにRENDAT.DAT（練習データ.データ、安直な命名）で、GO！

MS-DOSは、自分自身が4つのファイルをもっています。

CON　入力しているときは、キーボード。出力されるときは、画面。
PRN　プリンタ
AUX　RS-232C
NUL　ヌル

CONは、A>の状態のとき、直接フロッピーディスクに書き込むことができます。

このRENDAT.DATの中身を見るときには、TYPEコマンドを使いましたが、COPYコマンドで見ることができます。

```
A>COPY CON B:RENDAT.DAT
AAAAAAAA
BBBBBBBB
CCCCCCCC
^Z
        1 個のファイルをコピーしました.

A>
```

A>COPY B:RENDAT.DAT CON［リターン］

ドライブBのファイル、RENDAT.DATを画面にコピーします。さて？　どうなりましたか。TYPEコマンドと同じように画面に表示されましたよね。どこが、TYPEコマンドと違っていますか。TYPEコマンドと比べてください。

A>TYPE B:RENDATA.DAT［リターン］

```
A>COPY B:RENDAT.DAT CON
AAAAAAAA
BBBBBBBB
CCCCCCCC
        1 個のファイルをコピーしました.

A>
```

▲COPY B:RENDAT.DAT COM

```
A>TYPE B:RENDAT.DAT
AAAAAAAA
BBBBBBBB
CCCCCCCC

A>
```

▲TYPE B:RENDAT.DAT

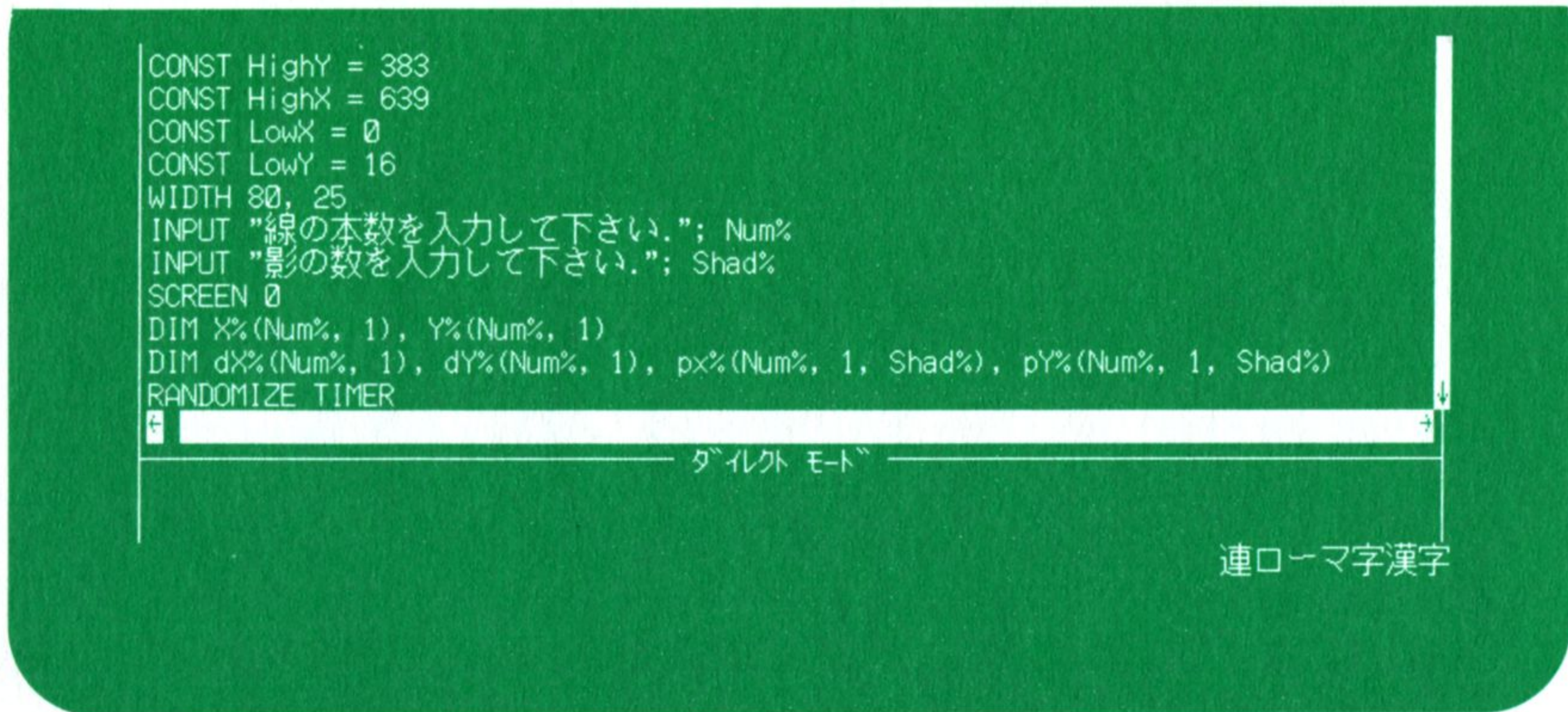
MS-DOSの画面の状態（A>）で、キーボードから入力した文字を、そのまま画面に表示します。入力が終わったことをコンピュータに教えるのは、[CTRL]＋[Z]キーです。

A>COPY CON CON［リターン］

適当にキーボードから文字を入力してみましょう。日本語を入力する場合には、[CTRL]＋[XFER]で、日本語入力状態にします。

```
CONST HighY = 383
CONST HighX = 639
CONST LowX = 0
CONST LowY = 16
WIDTH 80, 25
INPUT "線の本数を入力して下さい."; Num%
INPUT "影の数を入力して下さい."; Shad%
SCREEN 0
DIM X%(Num%, 1), Y%(Num%, 1)
DIM dX%(Num%, 1), dY%(Num%, 1), px%(Num%, 1, Shad%), pY%(Num%, 1, Shad%)
RANDOMIZE TIMER
```

ダイレクト モード

連ローマ字漢字

プリンタをもっている人は、CONをPRNに変えて、実行してみてください。

A>COPY B:RENDAT.DAT PRN［リターン］

で、ファイルの中身が、プリントアウトされます。
また、

A>COPY CON PRN［リターン］

で、簡易タイプライターに変身です。

第3章　プログラムのコンパイル

3.1　コンピュータへの命令と結果の表示

プログラミングとは、コンピュータを動かすための命令を、ユーザーである“あなた”が書いていくことです。そして、コンピュータが働いた結果を画面に表示させます。暗算で結果がわかる式で、コンピュータにたし算をやってもらいましょう。

コンピュータが正しく働いているかどうかを確認します。内容は、算数の1+2を計算（働かせて）させて、画面に表示させることとします。

```
PRINT 1+2 [リターン]
```

[SIFT]+[f･5]を押してください。Quick BASICのプログラムを入力した画面が切り換わって、3が表示されたでしょう。

これは、コンピュータに「1+2の結果を画面に表示せよ」という命令を与え、その命令によってコンピュータが働き結果を表示しました。

このように、コンピュータに与える命令はシンプルで明快です。これらの命令を組み合わせることでコンピュータが働くのです。この命令のことを、ステートメントといい、約150くらい用意してあります。また、たし算の+記号のような演算子

（えんざんし）やサイン、コサインなどの関数が、だいたい50ほどあります。これらを、一気に全部覚える必要はありませんし、不可能でしょう。

人は、自分のことを「おれ」、「ぼく」、「あたし」、「わたくし」、「拙者（せっしゃ）」などと言い、相手の人を呼ぶときにも、「きみ」、「あなた」、「おまえ」……などというように、いろいろな言い方ができますが、コンピュータには、PRINTの一言で、画面に表示したり、プリンタに出力したり、ディスクに書き込んだりします。そのため、“どんな”PRINTというところでツイとまどってしまいます。BASICの言語そのものは、PRINT文とINPUT文でおおかたの用事は間に合います。だ・か・ら、BASICは難しくありません。ただ、ほかの約束ごとがややこしいのです。

それでも、書き方、言い方には、一定のルールがあります。それを、本書を含め、多くの先輩が小さなプログラム例を発表していますので、少しずつ覚えて自分の言葉として身につけてください。

BASICを使って、自由にプログラムを組めるようになるには、早い人で約半年、普通でも1～3年くらいはかかるようです。気長にいきましょう。

1　変数という箱

画面に表示する命令はPRINT文を使って結果を表示させます。例では、1+2の結果を一時期、変数という箱の中にしまって、改めて表示させる方法を使います。この箱（変数）を使うとプログラムの実行中に、中身を自由に変えることができます。Quick BASICには、数字を入れる数値型の箱と文字を入れる文字列型の箱が用意されていて、どちらの箱を使うか区別する必要があります。この箱を区別することを型を宣言するといっています。

◎箱（変数）の名前のつけ方のルール

1．アルファベット、数字を使ってつける
2．最大は40文字以内
3．かならずアルファベットで始まる
4．予約語は使えない

◎数値型変数名

```
HAKO=1+2
PRINT HAKO
```

また、変数同士を式として使うこともできます。

```
A=2
B=3
C=A+B      <-----Cの箱にAの数字とBの数字をたした結果を入れます。
D=A*B
E=C+D
PRINT C    <------たし算の結果5が画面に表示されます。
PRINT D    <------かけ算の結果6が画面に表示されます。
PRINT E    <------CとDのたし算の結果11が画面に表示されます。
```

◎文字型変数名

アルファベットや数値を使った40文字以内の変数名の後ろに$をつけて、文字列は、”（ダブルクォテーション）で囲んで、数値型と区別します。

```
A$="Quick"
B$=" BASIC"
PRINT A$+B$  <-----Quick BASICと画面表示されます。
```

2　キーボードからのデータ入力

三角形の面積の求め方は、底辺×高さ÷2です。適当に、底辺の値と高さの値を入力すると、三角形の面積が画面に表示されるようにします。キーボードからの入力の命令は、INPUTです。

INPUTと画面表示のPRINT、加減乗除の演算子の組み合わせです。

```
1．REM　SPL001.BAS 三角形の面積プログラム
2．CLS
3．PRINT "▲▲▲三角形の面積▲▲▲"
```

```
4．INPUT "底辺＝ "; TEIHEN
5．INPUT "高さ＝ "; TAKASA
6．MENSEKI = TEIHEN * TAKASA / 2
7．PRINT "三角形の面積は "; MENSEKI; "cm2です。"
8．PRINT
9．END
```

Quick BASICには、行番号がつきません。プログラムの内容を説明するためにとりあえず、1～9の番号をつけました。

1．は、REM（レム文）といって、プログラムの注釈文を書きます。これは、プログラムの実行とは関係がなく、'（シングルコート）で書くこともできます。

2．実行画面をクリアします。CLSは、CLS 0、CLS 1、CLS 2とオプションをつけることができます。オプションの内容は、次のとおりです。

オプション	内容
なし	テキスト、グラフィック画面をクリアする
0	オプションなしと同じ
1	グラフィック画面だけをクリアする
2	テキスト画面だけをクリアする

クリアしたあと、カーソルは、画面の左端上段（ホームポジション）に移動します。

3．タイトルを画面に表示します。日本語入力状態にするのは、[CTRL]+[XFER]キーです。解除の場合も同じキーを使います。

4～5．キーボードからデータ入力を促します。""で囲まれた文字列がメッセージです。;（セミコロン）を忘れないように！

6．三角形の計算式です。

7．1行分何もないものを画面に表示させます（結果として、1行空白ができる）。

8．計算結果を表示します。

9．このプログラムの終了を意味します。

ここで、扱ったQuick BASICの命令語を整理してみると、わずかに5つです。

REM（あるいは　'）、CLS、PRINT、INPUT、END

暇なときに、Quick BASIC Statement & Function Reference（リファレンスマニュアル）を見て、内容を再確認してください。

プログラムのINPUTのあとにあるTEIHENやTAKASA、MENSEKIが変数です。この変数名は、前述のルールに添っていれば、TEIHENでもTEIHENDAでもTEIでも構いません。40字以内であれば、TEIHENDATEIHENDAOYABUNと八っあんの「口癖」でも変数名になります。もちろんAやBなどの1文字でもよいのですが、変数の中身を忘れてしまことがありますから、ある一定の長さで書いたほうが、あとあと自分のためになるでしょう。

3　見やすいプログラムの書き方

Quick BASICで、プログラムを書いていくときには、大文字で書いても小文字で書いてもかまいません。さらに、すべてのプログラムを改行した左端から書くことも、何らプログラムの実行には影響しません。今後、長いプログラムや複雑なプログラムを書くことを考えると、見やすいプログラムを書いたほうが、入力ミスや計算式のミスを発見するのに便利だと思います。

プログラムを見やすくするために、空白やタブを適当に入れて、先頭位置を右に下げます。通常、字下げは、キーボード上にある[TAB]キーを使ったTAB間隔で行ないます。Quick BASICでは、TAB間隔は、最初は4文字設定ですが、[V/表示]メニューの[O/オプション]で変更できます。

それと、プログラムの最初には、かならず注釈文をつけるように習慣つけましょう。時間がたつと、自分で作ったプログラムの内容を忘れてしまいます。

最初のサンプルプログラムは、適当に底辺と高さの数値を入力すると三角形の面積の結果を表示するプログラムです。REM文の頭のSPL001.BASは、プログラムのファイル名です。

```
'SPL001.BAS 三角形の面積プログラム
CLS
    PRINT "▲▲▲三角形の面積▲▲▲"
        INPUT "底辺＝ "; TEIHEN
        INPUT "高さ＝ "; TAKASA
            MENSEKI = TEIHEN * TAKASA / 2
    PRINT "三角形の面積は "; MENSEKI; "cm2です。"
    PRINT
END
```

```
▲▲▲三角形の面積▲▲▲
底辺＝ ? 20
高さ＝ ? 30
三角形の面積は  300 cm2です。
```

〈練習問題１〉

台形の面積を求める式を作ってみましょう。

台形の計算式は、（上底＋下底）＊高さ／2です。

```
 F:ファイル  E:編集  V:表示  S:検索  R:実行  D:デバッグ  O:オプション       H:ヘルプ
                              DAIKEI.BAS
'練習問題１　台形の面積　DAIKEI.BAS
CLS
    PRINT "□□□台形の面積□□□"
        INPUT "上底＝ "; JOUTEI
        INPUT "下底＝ "; KATEI
        INPUT "高さ＝ "; TAKASA
            MENSEKI = (JOUTEI + KATEI) * TAKASA / 2
    PRINT "台形の面積は "; MENSEKI; "cm2です。"
    PRINT
END
```

3.2　入力の繰り返しと分岐

三角形の面積を求める仕事をコンピュータにやってもらいましたが、たった1回きりです。何回か繰り返したいけど、毎回、[SHIFT]+[f･5]を押して繰り返すの非常にばかばかしいと思いませんか？　PRINTとINPUTだけで結果は出せるのですが、もっと楽をしたいということで、リファレンスマニュアルを調べると、繰り返しの命令語が、4種類用意してあります。

1．FOR...NEXT　ある決められた回数を繰り返すのに適しています。
2．WHILE...WEND　ある条件が真である間、繰り返すのに適しています。
3．DO WHILE(UNTIL)...LOOP　最初の条件が真（偽）の間、繰り返し実行します。
4．DO...LOOP WHILE(UNTIL)　繰り返し後の条件が真（偽）になるまで実行します。

また、決められた回数の繰り返しでないときには、ある一定の条件が満たされたとき、繰り返しのプログラムから脱出をはかる必要があります。脱出を計ることを「分岐」といい、分岐を行なうための計算式を「条件式」と言います。これらはBASIC言語ではありません。コンピュータを扱う上での約束ごとです。

これらも、ひとつ、ひとつ理屈で覚えたのではたいへんです。ここではそんなものもあるという程度でかまいません。

条件式に使う演算子には、

1．算術演算子
2．文字列演算子
3．関係演算子
4．論理演算子

と、いうものが用意されています。

演算子というからわかりにくいのかも知れません。数式の計算記号だと覚えていてください。

１．は、＋（加算）、－（減算）、／（除算＝わり算）、＊（乗算＝かけ算）
２．は、＋（連結）のみ
３．は、＝（イコール）、＜（大なり）、＞（小なり）、＜＝、＞＝、＜＞
４．は、AND、OR、NOT

1　三角形の面積を10回求める

すでに、繰り返す回数が決まったプログラムです。ここでは、10回になっていますが、I=1 TO 10の10の数字を別の数値に変えてやると、変えた数値だけ繰り返します。このとき、数値変数Iの中に1から順番に数字を入れていってカウントしていますが、変数名は、前述の条件にあっておれば、KAI、NOでも構いません。ちなみに、Iは、FORTRANの時代のカウンター変数の名残りです。余計なことですが、私はFORTRANを知りません。

```
 F/ファイル  E/編集  U/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                                  SPL002.BAS
' SPL002.BAS 三角形の面積プログラム10回繰り返し
CLS
  FOR I = 1 TO 10  '  <-----------------------------|
                                                   '|
    PRINT "▲▲▲三角形の面積▲▲▲"                    '| この間で10回繰り返
        INPUT "底辺= "; TEIHEN                     '| されます。
        INPUT "高さ= "; TAKASA                     '|
            MENSEKI = TEIHEN * TAKASA / 2          '|
    PRINT "三角形の面積は "; MENSEKI; "cm2です。"   '|
    PRINT                                          '|
  NEXT                 '  <-------------------------|

END

─────────────────── ダイレクト モード ───────────────────

<SHIFT+F1=ヘルプ> <F5=継続> <F9=ブレークポイント設定> <F8=トレース>   C  00006:009
```

```
▲▲▲三角形の面積▲▲▲
底辺＝ ? 10
高さ＝ ? 20
三角形の面積は  100 cm2です。

▲▲▲三角形の面積▲▲▲
底辺＝ ? 30
高さ＝ ? 40
三角形の面積は  600 cm2です。

▲▲▲三角形の面積▲▲▲
底辺＝ ? 50
高さ＝ ? 60
三角形の面積は  1500 cm2です。

▲▲▲三角形の面積▲▲▲
底辺＝ ? 70
高さ＝ ? 80
三角形の面積は  2800 cm2です。

▲▲▲三角形の面積▲▲▲
底辺＝ ?
```

2　真（しん）と偽（ぎ）

繰り返しの回数が最初からわかっていて、なおかつ固定されているときに使う場合はFOR･･･NEXTがよいのですが、途中で「～だったらヤメーた！」という場合には、FOR･･･NEXTは適していません。

つまり、「10回繰り返したらヤーメた！」ならば、FOR...NEXTで間に合っても、「底辺の値に0を入れたらヤーメた！」だとか、「底辺と高さの値が同じだったらヤメーた！」などと、ある条件が整ったときにプログラムの繰り返しをやめる場合には、条件（判断）式と演算子（えんざんし）を使って終了させることになります。

そのためには、真（･･･の条件が正しかったら）と偽（･･･の条件と合わなかったら）ということを知る必要があります。

そこで、条件（判断）式と演算子を使います。特に、関係演算子と論理演算子を式の中で使って、結果を真(-1）と偽（0）で判断します。

ここでは、Quick BASICの式と演算子の一覧を紹介します。

◎算術演算子

数値データを加工するための演算記号です。

算術演算子	意味	式	結果
^	ベキ乗	3^2	9
＊	乗算	3＊2	6
／	除算	8／2	4
¥	整数除算	10/3	3（余りの端数を切り捨て）
MOD	剰余	9 MOD 2	1（余りだけを出す）
＋	加算	2+3	5
－	減算	3-2	1

◎文字列演算子

文字データを加工する演算記号です。

算術演算子	意味	式	結果
＋	連結	"Quick"＋" BASIC"	Quick BASIC

◎関係演算子

同種の式の答えを比較して、真（-1）と偽（0）で結果を返します。

算術演算子	意味	式	結果
＝	等しい	5=（10/2)	-1（真）
＜＞	等しくない	5＜＞(10/2)	0（偽）
＜	小なり	5＜10	-1
＜＝	同じか小なり	10＜＝5	0
＞	大なり	10＞5	-1
＞＝	同じか大なり	5＞=10	0

◎論理演算子

関係演算子の組み合えあせによる論理を求め、真（-1）と偽（0）で結果を返します。論理演算子には、次のようなものがあります。

AND	論理和
OR	否定

NOT　　　　論理積
XOR　　　　排他的論理和
EQV　　　　同値
IMP　　　　抱合

Quick BASICで使われる演算子はこれですべてです。これらを組み合わせることで、判断がなされます。

実践的な使い方として、"Y"あるいは"y"、あるいは"ﾐ"だったらの例です。

```
IF  ANS$="Y" OR ANS$="y" OR ANS$="ﾝ" THEN
PRINT "YES"
ELSEIF ANS$="N" OR ANS$="n" OR ANS$="ﾐ" THEN
PRINT "NO"
ENDIF
```

これは、文字変数ANS$に、大文字、小文字、カナを問わず、Yが入力されたら「YES」、そうでなくANS$にNが入力されたら「NO」を画面に表示します。”ﾝ”と”ﾐ”は、PC-9801のキーボードのYとNを見てください。カナ文字入力の状態になっていても、大丈夫なように対応したものです。

関係演算子と論理演算子を使ったプログラムです。

この中に、IF（もし）...THEN（だったら）...ELSEIF（違って、もし）という判断式が多用されています。判断式の終了は、END IFです。

```
'SPL003.BAS「人と動物」論理演算子
CLS
  PRINT "質問を番号でお答えください"
        INPUT "性別を教えてください　男=1　女=2　どちらも違う=3　"; sex
        INPUT "あなたは靴を履きますか 履く=1　履かない=2　", kutu
        INPUT "あなたは4足であるきますか　歩く=1　歩かない=2　", aruki
  IF sex = 1 AND kutu = 1 AND aruki = 2 THEN
        PRINT
        PRINT "あなたは、立派な男性です"
  ELSEIF sex = 2 AND kutu = 1 AND aruki = 2 THEN
```

```
        PRINT
        PRINT "あなたは立派な女性です"
    ELSEIF sex = 3 THEN
        PRINT
        PRINT "生物ではありませんね"
    ELSEIF sex <> 3 AND kutu = 2 OR aruki = 1 THEN
        PRINT
        PRINT "お住まいは、多摩動物園のオリの中の方？"
    END IF
END
```

ここでは、靴を履いて2本足で歩く男は＝立派な男性

靴を履いて2本足で歩く女は＝立派な女性

性別のないのは＝生物ではない

最後に、性別の区別があって、靴を履いていても、履いていなくても4足で歩くのは＝動物？

入力された3つの条件を組み合わせて、それぞれELSEIFで結果を出しています。プログラムは上から順番に実行されるので、ELSEIF sex=3からの3行をEND IFの前にもってくると、性別のない4足で歩くのも多摩動物園の動物になってしまいます。

3　合計と平均値を出す

数値を好きなだけ入力して、-999を入力すると、データ入力を終了して、件数、合計、平均値を表示します。プログラムの繰り返しは、『入力された数値が-999でなかったら繰り返せ』ということです。

```
'SPL004.BAS 合計と平均値  END=-999
CLS
PRINT "◆◆◆合計と平均値  END=-999◆◆◆"
WHILE ANS <> -999                 'ANSが-999だったら繰り返し終了
        Sum = Sum + ANS           'Sumの中にANSの値を貯える
        N = N + 1                 '回数
```

```
    PRINT N;                        '回数を画面表示
        INPUT "件目＝ "; ANS        '数値を入力
WEND                                '繰り返す
    PRINT
    PRINT "回数＝ "; N - 1
    PRINT "合計＝ "; Sum
    PRINT "平均＝ "; Sum / (N - 1)
 END
```

```
◆◆◆合計と平均値 END=-999◆◆◆
 1 件目＝ ? 1
 2 件目＝ ? 2
 3 件目＝ ? 3
 4 件目＝ ? 4
 5 件目＝ ? 5
 6 件目＝ ? 6
 7 件目＝ ? 7
 8 件目＝ ? 8
 9 件目＝ ? 9
 10 件目＝ ? -999

回数＝  9
合計＝  45
平均＝  5

何かキーを押してください
```

4　最大値、最小値を求める式の追加

合計と平均値のプログラムに、判断式　IF...THEN　を追加してやることで、最大値、最小値も同時に計算できます。このとき、プログラムの終了用に-999を使っていますので、最小値を求めるときには、-999を最小値としないように注意する必要があります。

このプログラムを見てわかるように、前述の合計と平均のプログラムを拡張させたものです。まず、核となるプログラムを作って、徐々に拡張していき、やがて最

終目的のプログラムに仕上げる方法が初心者にとってはとっつきやすいでしょう。

```
'SPL005.BAS 合計と平均値と最大・最小値  END=-999
CLS
max = -9999
min = 99999
    PRINT "◆◆◆合計と平均値と最大・最小値  END=-999◆◆◆"
WHILE ans <> -999                    'ANSが-999でなかったら繰り返し終了
        Sum = Sum + ans              'Sumの中にANSの値を貯える
        N = N + 1                    '回数
    PRINT N;                         '回数を画面表示
        INPUT "件目＝ "; ans         '数値を入力
  IF max < ans THEN max = ans
  IF min > ans AND ans <> -999 THEN min = ans 'ansの値がminよりも小さくて、
                              'なおかつ-999でなかったらminの中にansの値を代入する
WEND                                 '繰り返す
    PRINT
    PRINT "回数＝ "; N - 1
    PRINT "合計＝ "; Sum
    PRINT "平均＝ "; Sum / (N - 1)
    PRINT "最大値＝ "; max
    PRINT "最小値＝ "; min
END
```

```
 ◆◆◆合計と平均値と最大・最小値  END=-999◆◆◆
  1 件目＝ ? 1
  2 件目＝ ? 2
  3 件目＝ ? 3
  4 件目＝ ? 4
  5 件目＝ ? 5
  6 件目＝ ? 6
  7 件目＝ ? 7
  8 件目＝ ? 8
  9 件目＝ ? 9
  10 件目＝ ? -999

 回数＝  9
 合計＝  45
 平均＝  5
 最大値＝  9
 最小値＝  1
```

3.3　V4.5での実行用ファイルの作成

いままで、実行してきたプログラムは、インタプリタといって、プログラムが書かれている順番に実行させてきました。その結果、実行においてミスがないことがわかり、プログラムファイルとして保存してあります。このファイルのことをソースプログラム、あるいはソースリストといって、実行用のプログラムと区別して呼んでいます。これまで、MS-DOSの実行ファイルにしなかったのは、プログラムが小さくコンパイルするメリットがなかったからです。

それでは、コンパイルの練習として、「合計と平均値と最大・最小値」のプログラムをQuick BASIC V4.5でコンパイルします。

1　ソースプログラムの読み込み

すでに、保存してあるソースリストをQuick BASICに読み込みます。

1．Quick BASICを起動して、[GRPH]キーを押します。
2．カーソルがメインメニューバーに移動したことを確認します。
3．[F/ファイル]メニューからプルダウンメニューを開きます。
4．［↓］キーで[O/読込...]を選択して、[リターン]キーを押すか、項目の頭文字[O]キーを押します（次ページ上図）。
5．画面が切り換わり、ファイル名の入力を聞いてきます。
6．ドライブがA:¥になっているときは、ファイル名のところに、いきなりB:［リターン］と入力します。
7．ドライブの指定がB:¥に変わり、拡張子がBASのファイルが表示されます。
8．[TAB]キーを押して、ファイル一覧のウインドウにカーソルを移動させます。
9．カーソル移動キーでカーソルを移動させ、目的のソースファイル名を反転させます。

10.　［リターン］キーで、プログラムが読み込まれます（下図）。

```
F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション      H/ヘルプ
                         SPL006.BAS
N/新規
O/読込...
A/名前を変えて保存...      ------タイマーセット
                          ------スタート時間の表示 "00:00:00"
P/印刷...
                          --|Iを1万回      |Iが1万回繰り返すのを
X/終了                      |繰り返す      |20回繰り返させる
NE
    PRINT          '<------------------画面上で1行改行
    PRINT TIME$    '<------------------実行終了時間の表示

                       ダイレクト モード
F1=ヘルプ | プログラムを読み込みます                    |   C  00001:001
```

```
F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション      H/ヘルプ
                         SPL006.BAS
'SPL                       O/読込
CLS   ファイル名を設定してください

FOR   N/ファイル名: SPL004.BAS

      D:¥ITI03
NEXT             F/ファイル                    D/ディレクトリ
      DAIKEI.BAS                               ..
      SPL001.BAS                               [-A-]
      SPL002.BAS                               [-B-]
      SPL003.BAS                               [-C-]
      SPL004.BAS                               [-D-]
      SPL005.BAS

                         < 確認 >  < 取消 >  < H/ヘルプ >
F1=ヘルプ  RETURN=実行  ESC=取消  TAB=次項目  カーソルキー=選択  |  C  00001:001
```

<試してみよう>

ソースプログラムは市販のワープロソフトで作れる

MS-DOSのテキストファイルで保存ができるワープロソフトであれば、Quick BASICのソースリストを作ることができます。

一太郎でプログラムを入力して、拡張子に.BASをつけて、保存するだけでOKです。プログラムを入力するときに、プログラムのリファレンスの間の空白が全角にならないように注意します。アルファベットを含め、全角で入力されるとエラーになります。

Mifesは、このようなソースプログラムを入力するために作られたもので、ワープロソフトと区別するために、エディタと呼んでいます。

2　実行用ファイルを作る

実行用ファイルを作ります。今までは、[SIFT]＋[f･5]でプログラムが実行されましたが、これは、Quick BASICを起動しているときにしか実行できませんでした。しかし、実行用ファイルを作れば、Quick BASICのシステムを毎回起動させる必要がなくなり、MS-DOSが起動した状態でプログラムの実行ができます。

この実行ファイルを作ることをコンパイルといいます。

1．[GRPH]キーでメインメニューバーへカーソルを移動させます。
2．[R/実行]メニューを選択します。
3．プルダウンメニューをひらき、[X/EXEファイル作成...]を選択します。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
'SPL005.BAS 合計と平均値と最大・  S/スタート                SHIFT+F5
CLS                               R/リスタート
max = -9999                       N/継続                          F5
min = 99999
    PRINT "◆◆◆合計と平均値と    X/実行ファイル作成...
WHILE ans <> -999
        Sum = Sum + ans          'Su
        N = N + 1                '回数
    PRINT N;                     '回数を画面表示
        INPUT "件目= "; ans      '数値を入力
  IF max < ans THEN max = ans
  IF min > ans AND ans <> -999 THEN min = ans 'ansの値がminよりも小さくて、
                   'なおかつ-999でなかったらminの中にansの値を代入する
WEND                             '繰り返す
    PRINT
    PRINT "回数= "; N - 1
    PRINT "合計= "; Sum
    PRINT "平均= "; Sum / (N - 1)
──────────────────────── ダイレクト モード ────────────────────────

F1=ヘルプ ｜ 実行ファイル(EXE)を作成します              ｜   C  00001:001
```

4．EXEファイル名をつけます。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
                                 SPL005.BAS
'SPL005.BAS 合計と平均値と最大・最小値  END=-999
CLS
max = -9999
min = 9999 ┌──────────── X/実行ファイル作成 ────────────┐
    PRINT  │ 実行ファイル名とコンパイルスイッチを設定してください │
WHILE ans  │                                              │
        Su │ N/実行ファイル名: [SUM01.EXE              ]  │
        N  │                                              │
    PRINT  │ [X] D/デバッグ                               │
        IN │                                              │
  IF max < │                                              │ て、
  IF min > │                                              │ る
           │ < M/実行ファイル作成 >  < E/実行ファイル作成後終了 > │
WEND       ├──────────────────────────────────────────────┤
    PRINT  │                      < 取消 >  < H/ヘルプ >  │
    PRINT  └──────────────────────────────────────────────┘
    PRINT
    PRINT "
──────────────────────── ダイレクト モード ────────────────────────

F1=ヘルプ  RETURN=実行  ESC=取消  TAB=次項目  カーソルキー=選択  ｜  C  00001:001
```

5．次のようなメッセージが出て、コンパイルが終了（EXEファイルが作成された）しました。

メッセージの中のエラーが0になっていることに注意してください。

```
Run File [SPL005.EXE]: A:¥QB45¥SUM01.EXE
List File [NUL.MAP]:
Libraries [.LIB]: A:¥QB45¥LIB¥BCOM45A.LIB

BC D:¥ITI03¥SPL005.BAS/D/O/T/C:512;
Microsoft (R) QuickBASIC Compiler Version 4.50J
(C) Copyright Microsoft Corporation 1982-1989.
All rights reserved.

43244 Bytes Available
42060 Bytes Free

    0 Warning Error(s)
    0 Severe  Error(s)
LINK @~QBLNK.TMP

Microsoft (R) Overlay Linker  Version 3.69
Copyright (C) Microsoft Corp 1983-1988.  All rights reserved.

Object Modules [.OBJ]: /EX SPL005
Run File [SPL005.EXE]: A:¥QB45¥SUM01.EXE
List File [NUL.MAP]:
Libraries [.LIB]: A:¥QB45¥LIB¥BCOM45A.LIB
```

3　ファイルの実行

Quick BASICを終了させて、MS-DOSの状態にしてからファイルを実行させます。

EXEファイルの実行状態がわかりやすいように、新しくフォーマットしたディスクにSUM01.EXEをコピーして、ディスクの内容を確認後、SUM01［リターン］で実行しました。

```
A:¥SUM01
```

画面が切り換わり、合計と平均値と最大・最小のプログラムが実行されました。適当に、データを入力して終了の-999を入力してください。プログラムの終了後に、A>が表示されて、MS-DOSの状態になっていることがわかります。

```
◆◆◆合計と平均値と最大・最小値　END=-999◆◆◆
 1 件目= ? 1
 2 件目= ? 2
 3 件目= ? 3
 4 件目= ? 4
 5 件目= ? 5
 6 件目= ? 6
 7 件目= ? 7
 8 件目= ? 8
 9 件目= ? 9
 10 件目= ? -999

回数=  9
合計=  45
平均=  5
最大値=  9
最小値=  1

A:¥>

  C1   CU   CA   S1   SU    VOID  NWL  INS  REP  ^Z
```

3.4　V4.2で独立型実行用ファイルの作成

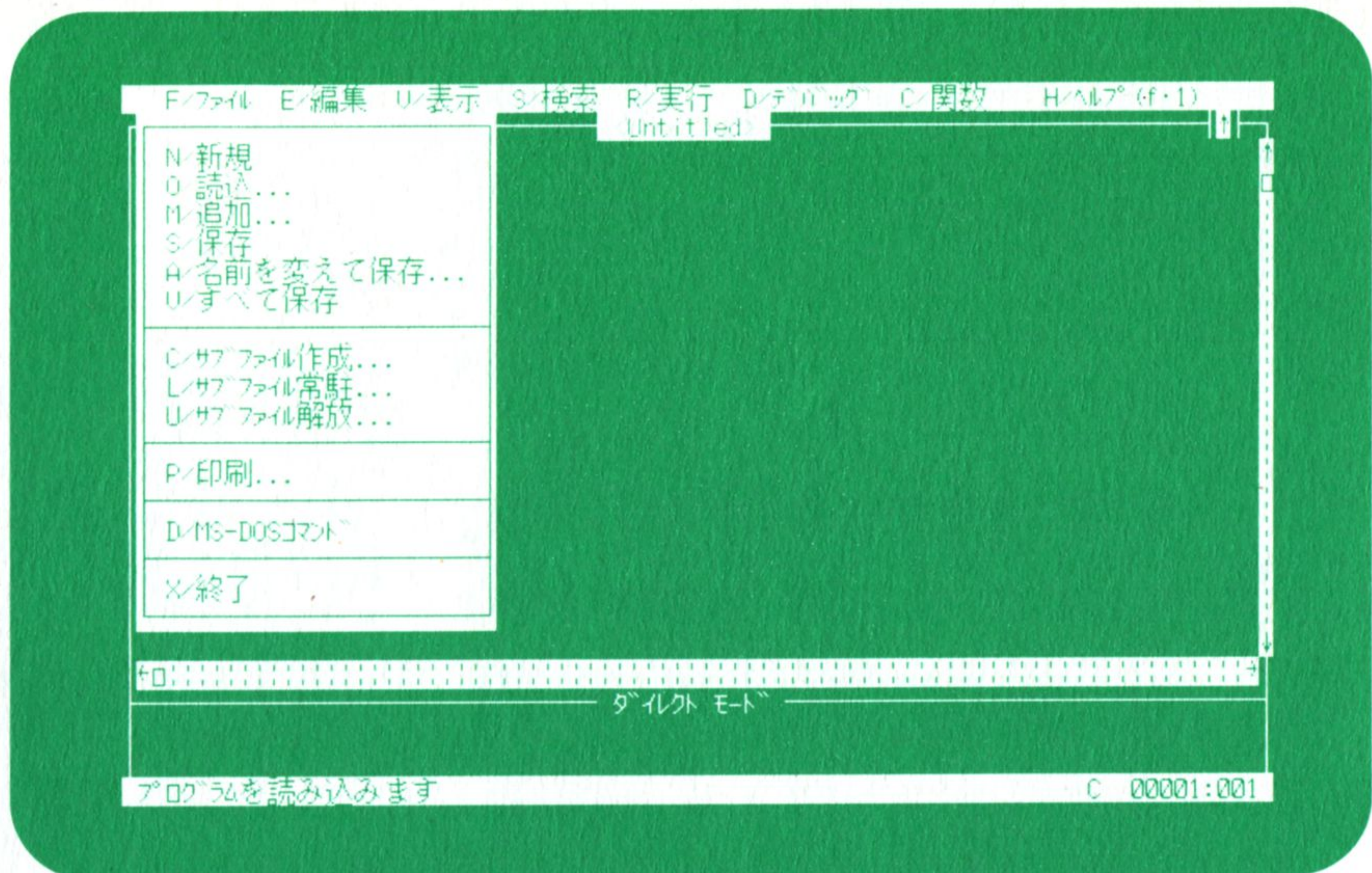
Quick BASIC V4.2では、ランタイム分離型実行ファイルと独立型実行ファイルの2種類のEXEファイルを作ることができるようになっています。ここでは、独立型実行ファイルを作ることにします。このファイルが完成すると、MS-DOSの起動画面からファイル名を入力するだけで、「合計と平均値と最大・最小値」のプログラムをスタートさせることができます。それでは、V4.5と同じように「合計と平均値と最大・最小値」のプログラムでコンパイルの練習をしましょう。実行ファイルの名前を、独立型はSUM02.EXE、分離型はSUM03.EXEとします。

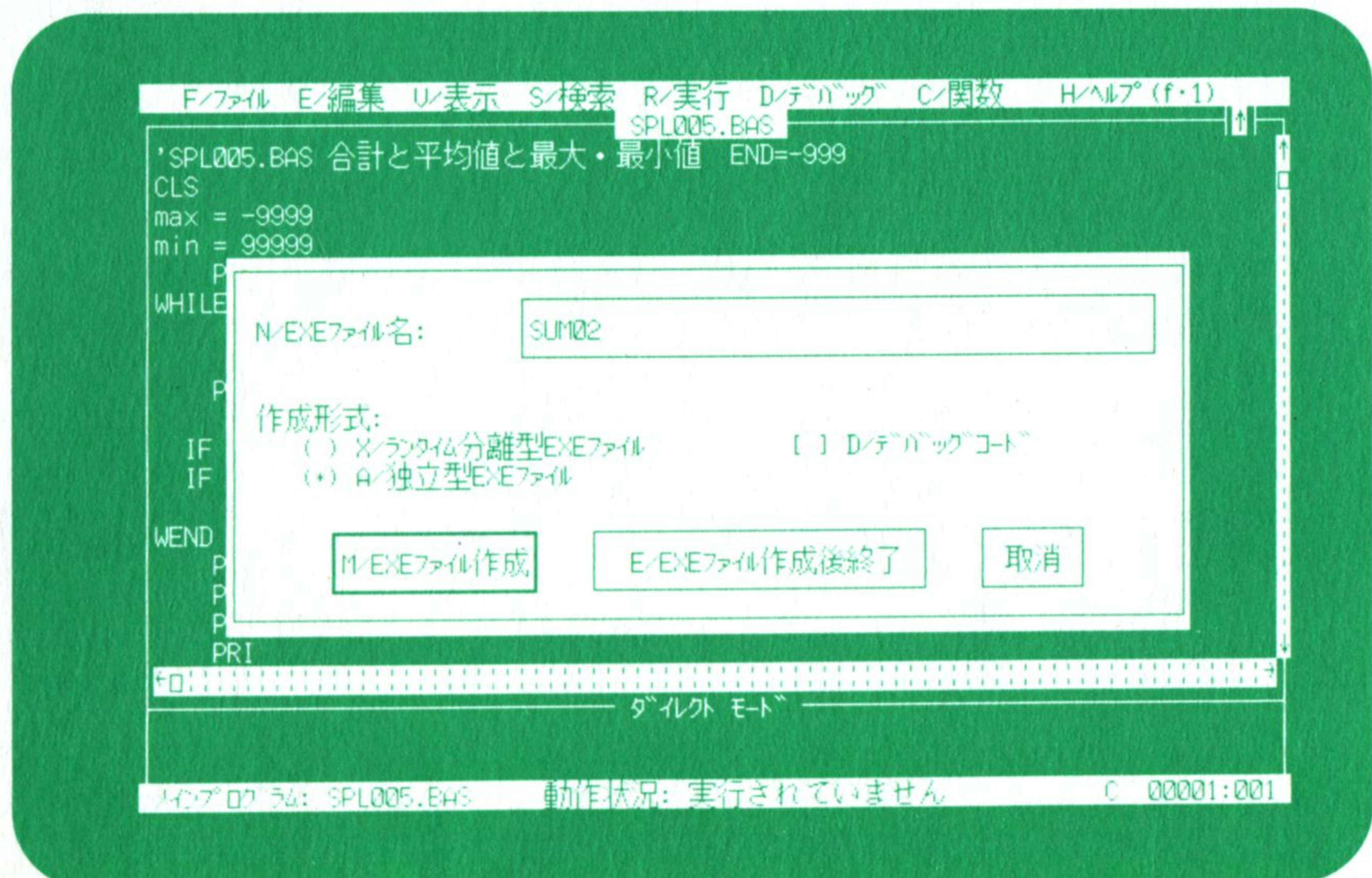
1 ソースプログラムの読み込み

すでに、保存してあるソースリストをQuick BASICに読み込みます。この間の操作は、V4.5と同じです。

1．Quick BASICを起動して、[GRPH]キーを押します。
2．カーソルがメインメニューバーに移動したことを確認します。
3．[F/ファイル]メニューからプルダウンメニューを開きます。
4．［↓］キーで[O/読込...]を選択して、[リターン]キーを押すか、項目の頭文字[O]キーを押します。
5．画面が切り換わり、ファイル名の入力を聞いてきます。
6．ドライブがA:¥になっているときは、ファイル名のところに、いきなりB:［リターン］と入力します。
7．ドライブの指定がB:¥に変わり、拡張子がBASのファイルが表示されます。
8．[TAB]キーを押して、ファイル一覧のウィンドウにカーソルを移動させます。
9．カーソル移動キーでカーソルを移動させ、目的のソースファイル名を反転させます。
10．[リターン]キーで、プログラムが読み込まれます。

2 独立型実行用ファイルを作る

1．[GRPH]キーでメインメニューバーへカーソルを移動させます。
2．[R/実行]メニューを選択します。
3．プルダウンメニューをひらき、[X/EXEファイル作成...]を選択します。

4．EXEファイル名を「SUM02」とつけます。

5．作成形式に2種類あるので、[TAB]キーを使って独立型EXEファイルを選択します。

6. 自動的にコンパイルされます。

メッセージの中のエラーが0になっていることに注意してください。

```
BC D:¥IT-B002¥SPL005.BAS/O/T/C:512;
Microsoft (R) QuickBASIC KANJI Compiler Version 4.20
(C) Copyright Microsoft Corporation 1982-1988.
All rights reserved.

43511 Bytes Available
42299 Bytes Free

     0 Warning Error(s)
     0 Severe  Error(s)
LINK /EX SPL005,A:¥QBASIC42¥SUM02.EXE,NUL,;

Microsoft (R) Overlay Linker  Version 3.65
Copyright (C) Microsoft Corp 1983-1988.  All rights reserved.
```

3.5　V4.2ランタイム分離型実行用ファイルの作成

ランタイムモジュールを分離したランタイム分離型実行用ファイルの作成を行ないます。コンパイルしたEXEファイルを実行させるためには、ランタイムモジュールBRUN42A.EXEを同一ディスク内にコピーしてやる必要があります。

1　ソースプログラムの読み込み

ここまでの操作は、V4.5あるいは、V4.2の独立型実行用ファイルの作成と同じです。

2　ランタイム分離型EXEファイルを作る

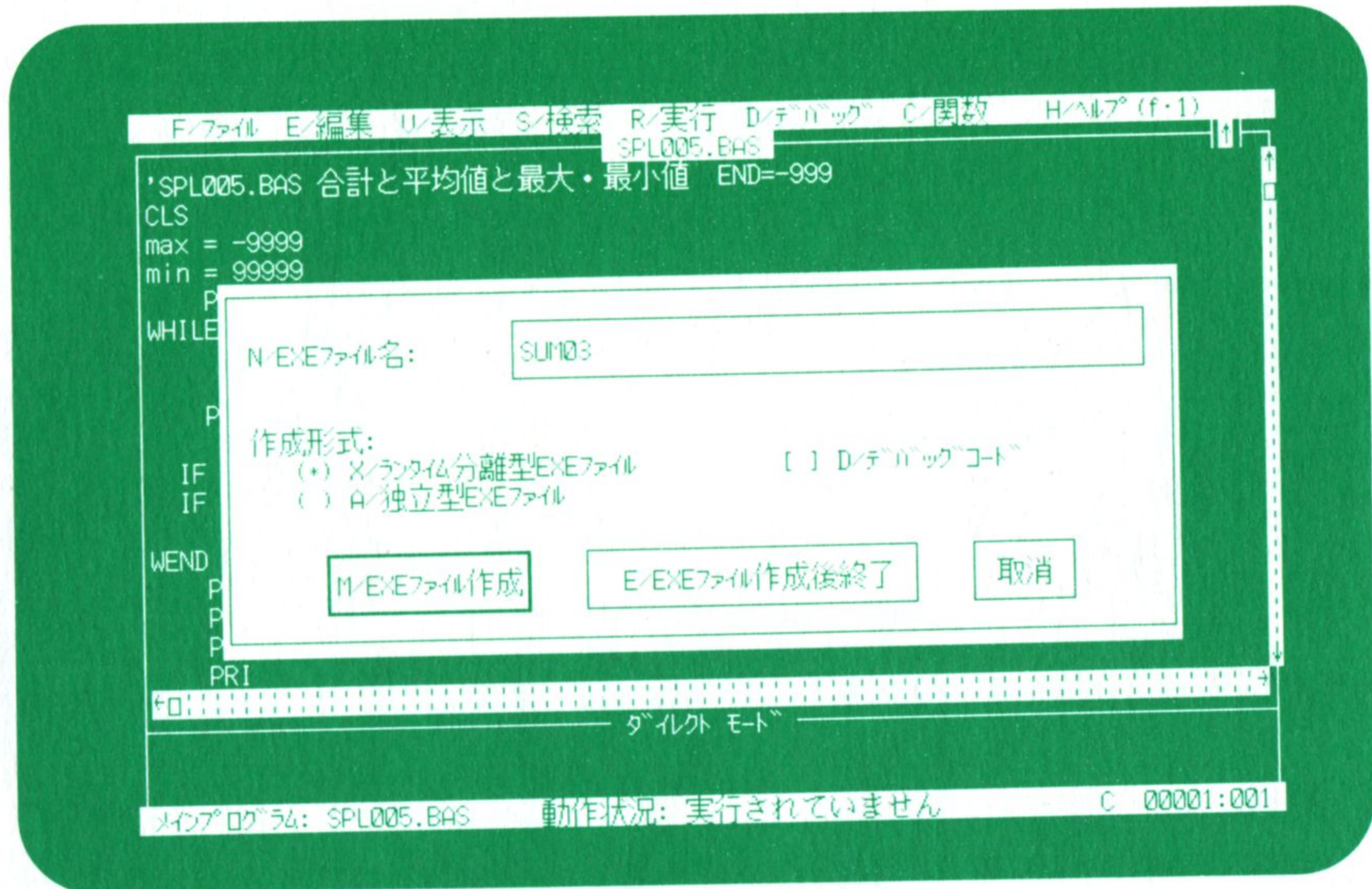
1．[GRPH]キーでメインメニューバーへカーソルを移動させます。

2．[R/実行]メニューを選択します。

3．プルダウンメニューをひらき、[X/EXEファイル作成...]を選択します。

4．EXEファイル名を「SUM03」とつけます。

5．作成形式に2種類あるので、分離型EXEファイルを選択します。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数    H/ヘルプ(f･1)
                              SPL005.BAS
'SPL005.BAS 合計と平均値と最大・最小値  END=-999
CLS
max = -9999
min = 99999
   P
WHILE    N/EXEファイル名:         SUM03

   P
         作成形式:
  IF        (•) X/ランタイム分離型EXEファイル           [ ] D/デバッグコード
  IF        ( ) A/独立型EXEファイル

WEND
   P        M/EXEファイル作成    E/EXEファイル作成後終了    取消
   P
   P
   PRI
                         ダイレクト モード

メインプログラム: SPL005.BAS    動作状況: 実行されていません       C  00001:001
```

6. 自動的にコンパイルされます。

メッセージの中のエラーが0になっていることに注意してください。

```
BC D:¥IT-B002¥SPL005.BAS/T/C:512;
Microsoft (R) QuickBASIC KANJI Compiler Version 4.20
(C) Copyright Microsoft Corporation 1982-1988.
All rights reserved.

43511 Bytes Available
42299 Bytes Free

    0 Warning Error(s)
    0 Severe  Error(s)
LINK /EX SPL005,A:¥QBASIC42¥SUM03.EXE,NUL,;

Microsoft (R) Overlay Linker  Version 3.65
Copyright (C) Microsoft Corp 1983-1988.  All rights reserved.
```

3　分離型EXEファイルの実行

Quick BASICを終了させ、MS-DOSの状態からファイルを実行させます。

ディスクにSUM03.EXEをコピーして、ディスクの内容を確認後、SUM03［リターン］で実行します。

```
A:¥>SUM03
Input run-time module path:
Input run-time module path:
Input run-time module path: B:¥BIN¥BRUN24A
```

画面に、”Input run-time mdule path:　”とメッセージが表示されて、実行が中断しています。これは、「ランタイムモジュールのあるパスを入力してください」といっています。無視して［リターン］キーを入力すると、同じメッセージが繰り返され、プログラムは実行されません。

ドライブBにQuick BASICのワークディスクを入れて、B:¥BIN¥BRUN42A［リターン］と入力すれば、プログラムは実行されますが、これでは、Quick BASICが必要になり、手間のかかることは、[SIFT]＋[f･5]を実行したときと同じですね。

これでは、ツマラナイので、1枚のディスクで実行できるようにしましょう。

ドライブAに実行用ファイル、ドライブBにQuick BASICのワークディスクが入っているものとして、次のように入力します。

```
COPY B:¥BIN¥BRUN42A.EXE A:［リターン］
```

ドライブAにBRUN42A.EXEというランタイムモジュールがコピーされていることを確認して、あらためて、

```
SUM03［リターン］
```

と入力します。

これで、合計と平均値と最大・最小のプログラムが実行されます。

4　実行ファイルの大きさ

SUM03.EXEは、ランタイムモジュールを必要とするEXEファイル、SUM01.EXEは、V4.5でコンパイルした独立型EXEファイル、SUM02.EXEは、V4.2の独立型のEXEファイルです。同じプログラムでも大きさが約10倍の値がちがいます。

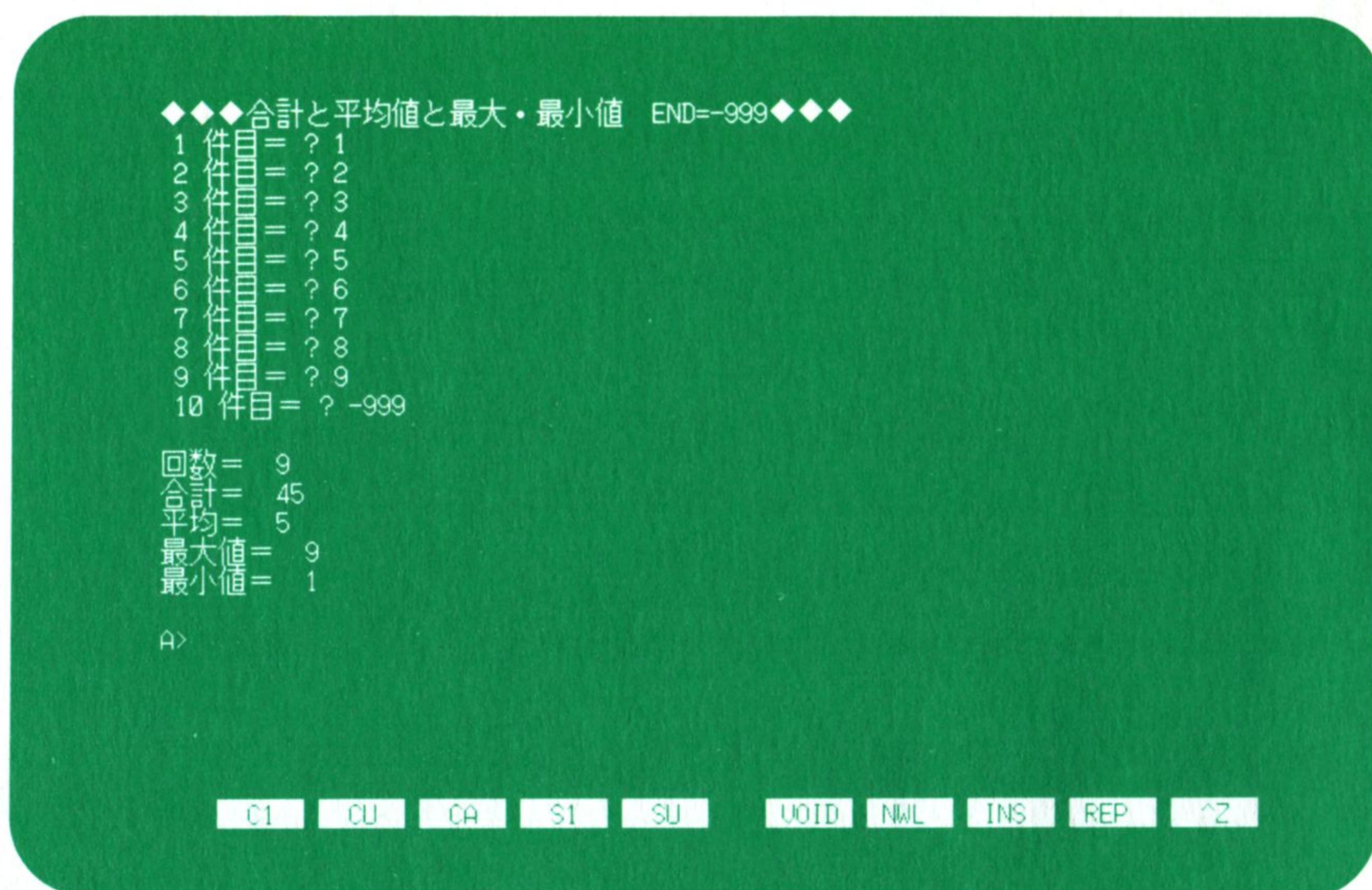
この大きさは、SUM01.EXE、SUM02.EXEは、BRUN42A.EXE（ランタイムモジュール）が取り込まれ、SUM03.EXEは、ランタイムモジュールが、分離したかたちの実行ファイルだと思ってください。

SUM01.EXE、SUM02.EXEは、このファイルだけでプログラムの実行ができます。

```
A>DIR

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM     24931  89-02-17    0:00
SUM01    EXE     46776  90-05-02   16:11
SUM02    EXE     42945  90-05-02   18:10
SUM03    EXE      4155  90-05-02   18:15
        4 個のファイルがあります.
  1034240 バイトが使用可能です.

A>

 C1  CU  CA  S1  SU    VOID NWL INS REP ^Z
```

▲実行ファイルの大きさ

▼SUM02の実行画面

```
◆◆◆合計と平均値と最大・最小値　END=-999◆◆◆
1 件目= ? 1
2 件目= ? 2
3 件目= ? 3
4 件目= ? 4
5 件目= ? 5
6 件目= ? 6
7 件目= ? 7
8 件目= ? 8
9 件目= ? 9
10 件目= ? -999

回数=  9
合計=  45
平均=  5
最大値=  9
最小値=  1

A>

 C1   CU   CA   S1   SU      VOID  NWL  INS  REP  ^Z
```

3.6　プログラムの実行速度

Quick BASICでは、3種類のプログラムの実行方法があります。実際に、プログラムを作って実行させるときのは、どの方法がいいのか調べてみることにします。

1　実行速度チェックプログラム

●実行速度チェック用「計算式」プログラム

コンピュータ本来の「計算」を中心にしたプログラムです。便宜上、「計算式」プログラムと呼ぶことにします。

```
'SPL006.BAS  KEISAN
CLS : Y = 2
  TIME$ = "00:00:00"          '<--------タイマーセット
        PRINT TIME$           '<--------スタート時間の表示" 00:00:00"
FOR CL = 1 TO 20              '<-----------------|
  FOR I = 1 TO 10000          '<----|Iを1万回     |Iが1万回繰り返すのを
                              '     |繰り返す     |20回繰り返させる
  NEXT                        '<----|            |
NEXT                          '<-----------------|

        PRINT                 '<------------------画面上で1行改行
        PRINT TIME$           '<------------------実行終了時間の表示
```

```
00:00:00
00:01:29




何かキーを押してください
```

2　実行速度チェック用「表示形」プログラム

コンピュータの計算は頭脳の中での実行。それを人間にたとえるならば、脳の指令を受けて、実際に手足を動かすものに近い動作をさせるため、画面に”o”を表示させていきます。このプログラムは、「表示形」と呼ぶことにします。

```
'SPL007.BAS HYOJI
CLS : Y = 2
  TIME$ = "00:00:00"
      PRINT TIME$
FOR CL = 1 TO 20
  FOR I = 1 TO 10000
IF I MOD 125 = 0 THEN PRINT "o"; '<------Iの値が125で割り切れたら”o”を
                                          '    表示します。
  NEXT
NEXT

      PRINT
      PRINT TIME$
```

```
00:00:00
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

00:02:56

何かキーを押してください
```

3　実行結果

実行速度を計ることは、いろいろな条件で変わってきます。たとえば、使っている機種が変われば違ってきますし、プログラムの組み方によっても変わってきます。当然、PC-9801もいろいろな機種がありますので、個々の機種によって、実行速度が変わるでしょう。この結果は、PC-9801Vm2でクロックを10MHzにセットし、前述のプログラムをQuick BASICV4.2で実行という条件の元で算出されたものです。

	インタプリタ	ランタイム型	独立型
計算式	1分30秒	57秒(33秒)	57秒(33秒)
表示形	2分55秒	1分48秒(1分7秒)	1分47秒(1分8秒)
差	1分22秒	51秒(31秒)	50秒(32秒)

(　)内のタイムはインタプリタと比べたときの差です。

結論として、EXEの実行ファイルはインタプリタの実行速度の約1.6倍。ほかの機種であれば、もっと差がでるのかどうかはわかりませんが、とにかく速くなっていることはわかります。

また、パソコン内部の計算式よりも画面に表示させるということに、以外と時間がかかっていることがわかります。これから、自分なりのプログラムを組むとき、不必要な画面表示を避けることで、速いプログラムを組むことができるヒントにはなります。

ランタイムモジュールを使うEXEファイルでも独立型でも差のないことがわかります。

なお、このプログラムは、パソコンの内部時計を使っていますので、内部時計がメタメタに狂ってしまいます。MS-DOSの起動時にちゃんとセットしなおしておいてください。

4　物理の公式を使う　＝物体の落下速度＝

ある高さから物が落下したときの時間を秒単位で計算します。公式によると、

等加速度直線運動の公式 $S=V_0t+\frac{1}{2}gt^2$

自由落下ですから　初速度$V_0=0$ですから、$S=\frac{1}{2}gt^2$

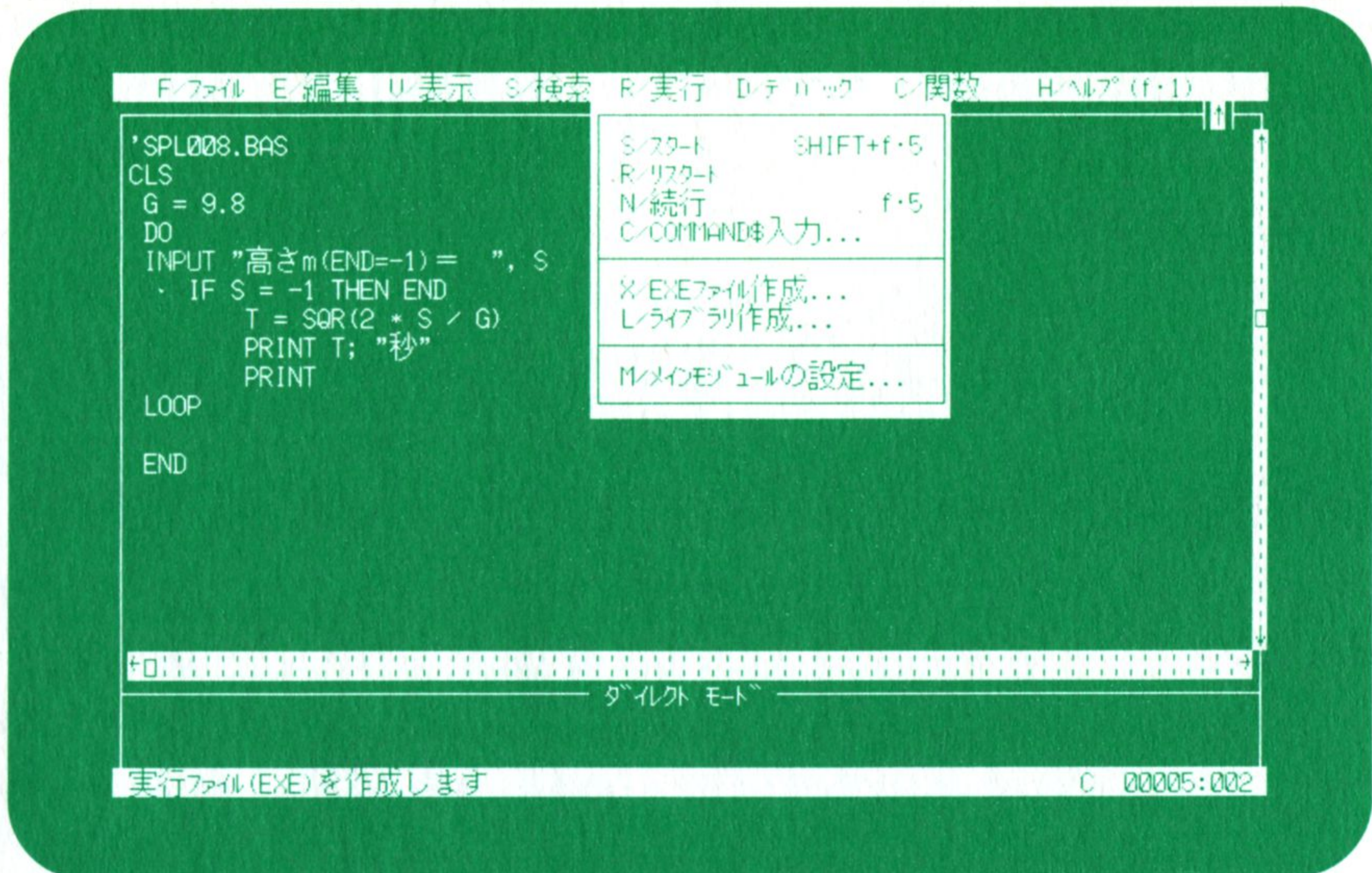
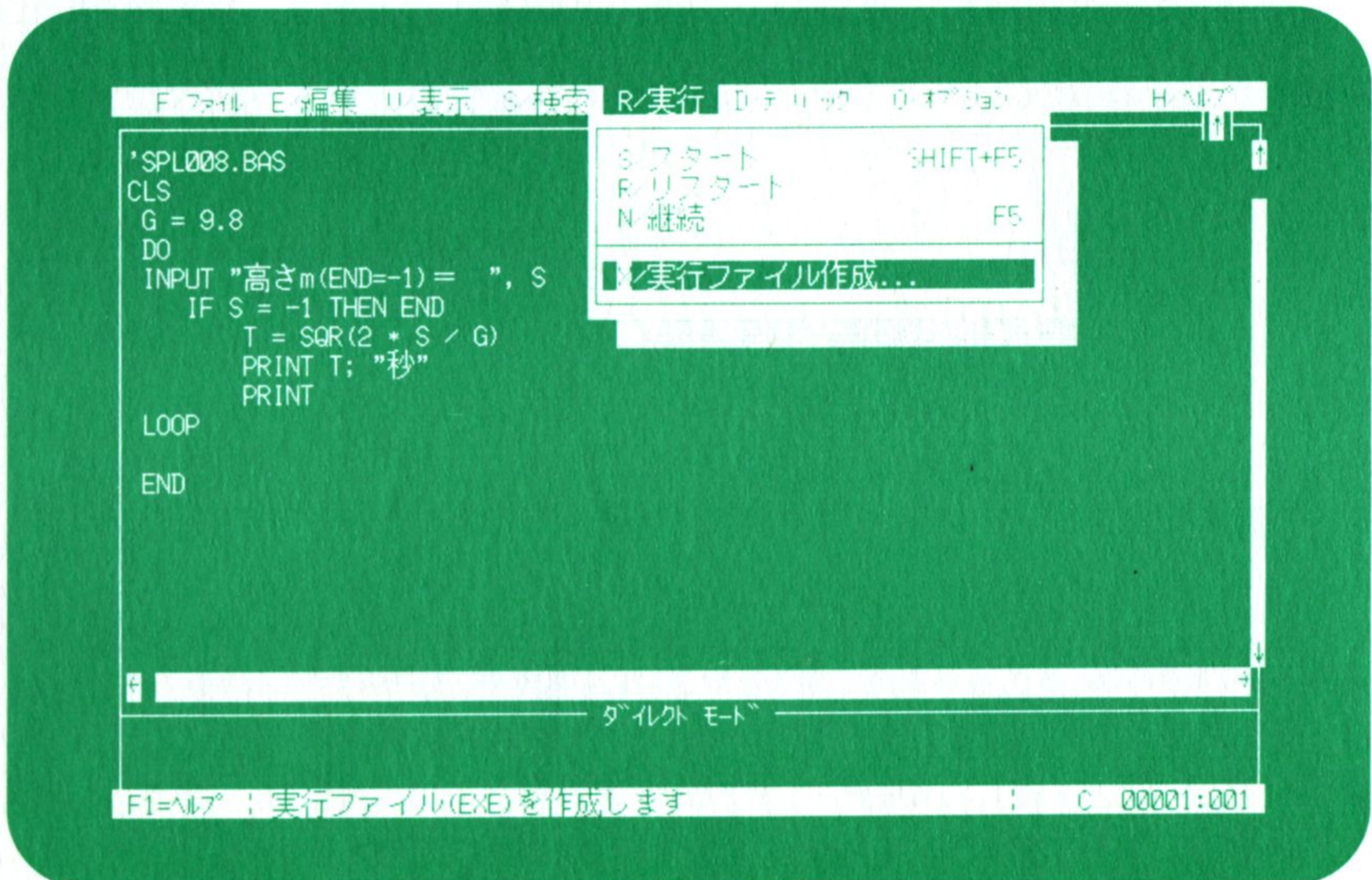
このことから、時間Tは、T=SQR(2*S/G)で求められます。

このプログラムを[SIFT]＋[f･5]で実行後、コンパイルをしてみましょう。

S:落下距離(m)　V0:初速度(m/秒)　t:時間(秒)　g:加速度(9.8m/秒2)

```
'SPL008.BAS
CLS
 G = 9.8
 DO
 INPUT "高さm(END=-1)＝  ", S
    IF S = -1 THEN END
        T = SQR(2 * S / G)
        PRINT T; "秒"
        PRINT
 LOOP

 END
```

富士山の高さは、3,776m。東京タワーは333m。風の抵抗がないものとして、それぞれからパラシュートをつけないで飛び降りると、富士山では、27.8秒、東京タワーからだと8.2秒ということになります。なんだか、物理学が身近なものになったと思いませんか。

```
高さm(END=-1)＝  3776
 27.7599 秒

高さm(END=-1)＝  333
 8.243736 秒

高さm(END=-1)＝
```

▲V4.2の実行用ファイル作成メニュー

▲V4.5の実行用ファイル作成メニュー

＜練習問題２＞

ボールの投げ上げ

初速度V0(m/秒)で投げ上げるとボールは、どこまで高く上がるでしょう。

そして、それは何秒後でしょう。初速度を入力して、結果を出してください。

高さhの式は、h=H^2/(2*9.8)

時間Tの式は、T=V0/9.8

第４章　画面表示の体裁を整える

4.1　セミコロンとカンマ

Quick BASICのプログラムの原点は、キーボードからの入力（INPUT）と画面への表示（PRINT）です。INPUTであれ、PRINTであれ、現在の状況がわかりやすいように、" "（ダブルクォーテション）を使ってメッセージをつけます。そして、そのあとに、変数名をつけます。それによって、キーボードからの入力の場合は、変数名の中に何らかのデータが取り込まれ、画面表示の場合は、変数の中のデータが画面に表示されます。この入力の状態や表示の状態を「,（カンマ）」と「;（セミコロン）」で変えることができます。

1　PRINTのカンマとセミコロン

まず、カンマもセミコロンもついていないプリント文を作り、Quick BASICの編集メニューからコピー命令を使います。

1．作ったプログラムを[SIFT]＋[↓]キーを使ってコピー範囲を決めます。指定された範囲は、反転します。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                              Untitled
CLS
A = 123
B = 456
C = 789
PRINT A
PRINT B
PRINT C
```

2．[GRPH]キーを押して、メインメニューバーへカーソルを移動します。

3．[E/編集]を選択して、プルダウンメニューを開き、[C/コピー]を選択します。

[CTRL]＋[INS]キーを使うこともできます。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                              Untitled
A = 123   T/カット      SHIFT+DEL
B = 456   C/コピー       CTRL+INS
C = 789   P/ペースト    SHIFT+INS
PRINT A
PRINT B
PRINT C
```

4．[リターン]キーを押したあと、コピー先にカーソルを移動させます。

5．[GRPH]キーを押して、再び、メインメニューバーにカーソルを移動させます。

6．[E/編集]を選択して、プルダウンメニューをひらきます。

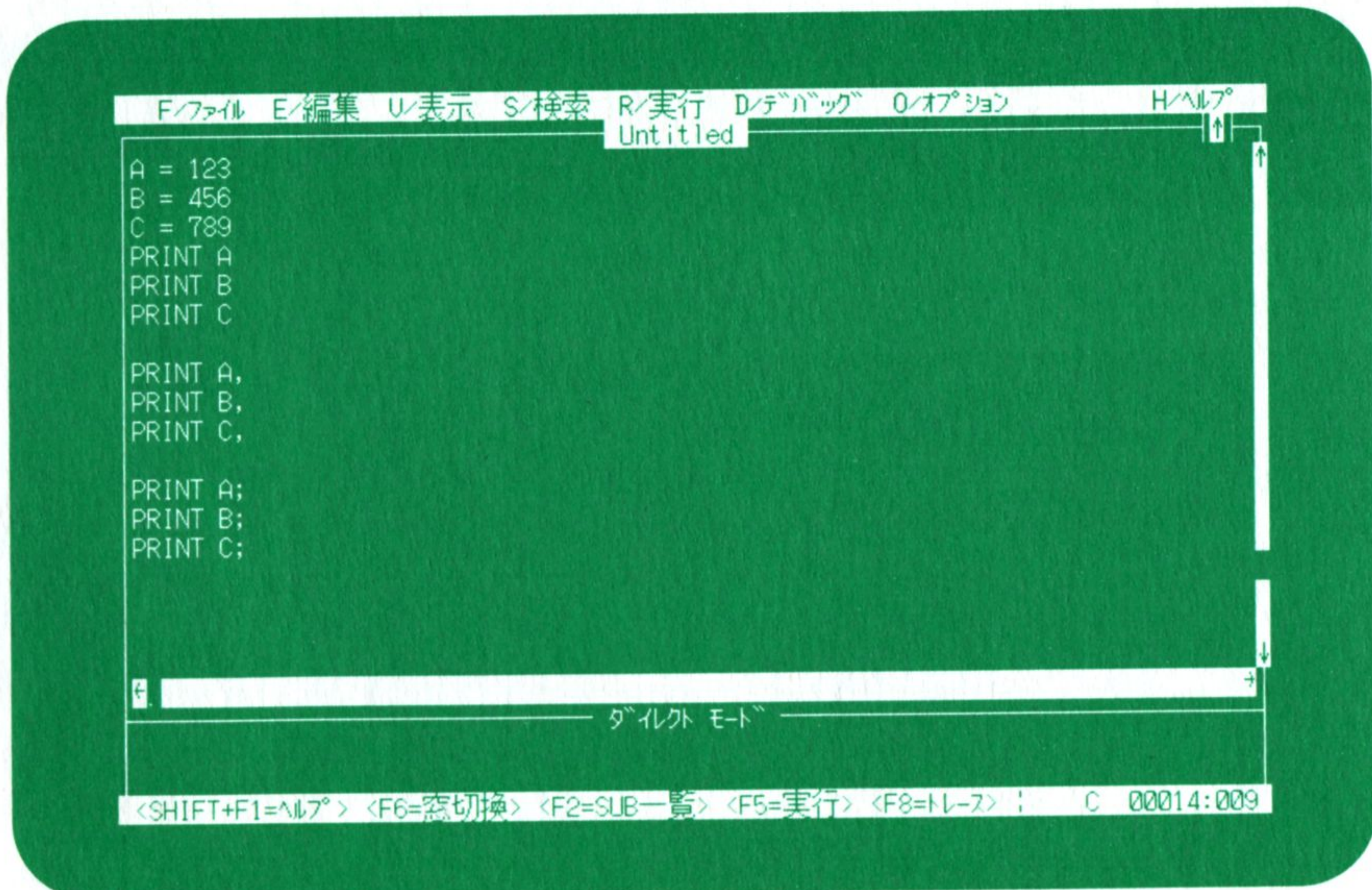
7．[P/ペースト]を選択して、[リターン]キーを押すと、範囲指定したプログラムがコピーされます。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                                   Untitled
A = 123   T/カット      SHIFT+DEL
B = 456   C/コピー       CTRL+INS
C = 789   P/ペースト    SHIFT+INS
PRINT A
PRINT B
PRINT C
```

9．次のコピー先にカーソルを移動させます。

10．ペーストのショートカットキー、[SIFT]＋[INS]を押すとコピーされます。ペーストは、次のコピーやカットの指示を行なうまで何度でも使うことができます。

11．コピーが終了したら、画面のように、それぞれのプログラムの後ろにカンマとセミコロンをつけます。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                                   Untitled
A = 123
B = 456
C = 789
PRINT A
PRINT B
PRINT C

PRINT A,
PRINT B,
PRINT C,

PRINT A;
PRINT B;
PRINT C;

                              ダイレクト モード

<SHIFT+F1=ヘルプ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=トレース> ¦    C  00014:009
```

さっそく、[SIFT]＋[f·5]で実行してみましょう。

変数A、B、Cの後ろに何もつけていないと1行ずつ改行されて表示されます。

```
123
456
789
```

次の表示は、,（カンマ）をつけた例です。画面1行に8文字間隔で表示されています。

```
123       456       789
```

最後は、1行に文章がつながっています。;（セミコロン）を使うと、プログラムが改行になっていても、ちゃんと1行で画面に表示します。

```
123  456  789
```

2　INPUTのカンマとセミコロン

名前をキーボードから入力するプログラムです。プログラムのコピーは、

1．[SIFT]＋[↓]で範囲指定

2．[CTRL]＋[INS]でコピー

3．コピー先へカーソルの移動

4．[SHIFT]＋[INS]でペースト

でした。次のプログラムは、INPUTとPRINTを使った例です。文字を入力しますので、変数名の後ろに$をつけます。

```
F/ファイル E/編集 V/表示 S/検索 R/実行 D/デバッグ O/オプション H/ヘルプ
Untitled
'INKORON.BAS
CLS

INPUT "1.名前を入力してください=  "; NAME1$
INPUT "2.名前を入力してください=  ", NAME2$
    PRINT
    PRINT "1の名前は (カンマ区切り表示) =  ", NAME1$
    PRINT "1の名前は (セミコロン 区切り表示) =  "; NAME1$

    PRINT "2の名前は (カンマ区切り表示) =  ", NAME2$
    PRINT "2の名前は (セミコロン 区切り表示) =  "; NAME2$

END
```

プログラムを実行します。[SHIFT]＋[f·5]でした。

INPUT文にセミコロンをつけると、?が表示されます。

```
1.名前を入力してください=  ? 遠藤謙一
2.名前を入力してください=  えんどうけんいち

1の名前は (カンマ区切り表示) =              遠藤謙一
1の名前は (セミコロン 区切り表示) =  遠藤謙一
2の名前は (カンマ区切り表示) =              えんどうけんいち
2の名前は (セミコロン 区切り表示) =  えんどうけんいち
```

3　カンマとセミコロンを組み合わせて表示

誕生日を聞いています。月を聞いてきたときには、?マークがあることに注意してください。また、表示を1列で行なうために、セミコロンがPRINT文で使われています。セミコロンでつなぐ表示の場合は、変数も入れることができるので、比較的美しい画面表示を行なうことができます。

```
F/ファイル E/編集 V/表示 S/検索 R/実行 D/デバッグ O/オプション H/ヘルプ
INKORON.BAS
'PRKORON.BAS
CLS

INPUT "生まれた月を入力してください=  ", MON$
INPUT "生まれた日を入力してください=  ", DAY$
    PRINT
    PRINT "あなたは"; MON$; "月"; DAY$; "日生まれですね。"

END
```

カンマとセミコロンを組み合わせた結果です。入力部と表示部をはっきりわけるのであれば、日にちのINPUT文のあとにPRINTを入れてやると画面上で、1行空きます。

```
生まれた月を入力してください=  5
生まれた日を入力してください=  31

あなたは5月31日生まれですね。
```

<試してみよう>

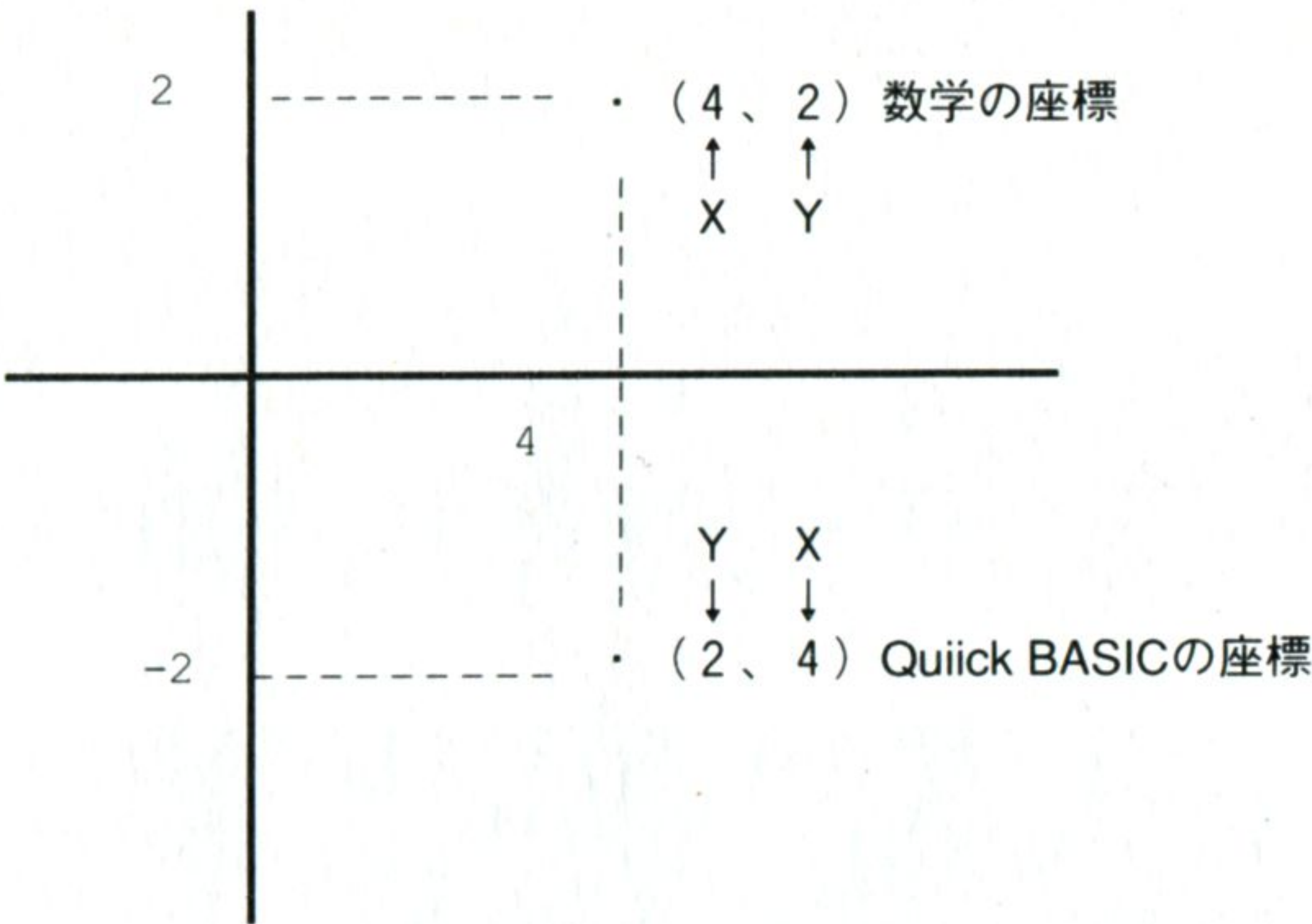
誕生日を聞いてきているプログラムでは、文字変数（変数名の後ろに$）を使っています。当然、数値入力ですから、数値変数でよいはず。変数名の$（4カ所）をはずして、プログラムを実行してみてください。表示がどのように変わりましたか。また月日に－（マイナス記号）をつけてみてください。

4.2　狙った位置に文字表示

画面に表示できる文字の数は、1行80字で25行です。これらの位置は、Y軸、X軸の交点を指定して、文字の表示を行なえば、画面の好きな位置に表示することができます。

1　画面のY軸、X軸

数学のグラフを作る場合のX軸、Y軸（Quick BASICの場合は、Y軸、X軸）は、次のようになっています。

2
・（4、2）数学の座標
X　Y
4
Y　X
−2
・（2、4）Quiick BASICの座標

画面上でのY軸、X軸は、画面の右端、上部になります。

1,1	1,2	1,3	1,4	...	1,79	1,80
2,1	2,2	2,3	2,4	...	2,79	2,80
3,1	3,2	3,3	3,4	...	3,79	3,80
24,1	24,2	24,3	24,4	...	24,79	24,80
25,1	25,2	25,3	24,4	...	25,79	25,80

2　画面中央にQuick BASIC

画面上のY軸、X軸の交点を表わすのにLOCATE文を使います。例題として、画面中央に“Quick BASIC”を表示してみましょう。

LOCATE文のあとに:（コロン）をつけて、PRINT文が続いています。このように、Quick BASICでは、複数の命令を1行につないで使うことができます。このスタイルのことをマルチステートメントといい、かならず、「:（コロン）」を使います。

```
F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                              SPL3-001.BAS
'SPL3-001.BAS
CLS
    LOCATE 12, 40: PRINT "Quick"
    LOCATE 13, 40: PRINT "BASIC"

END
```

```
Quick
BASIC
```

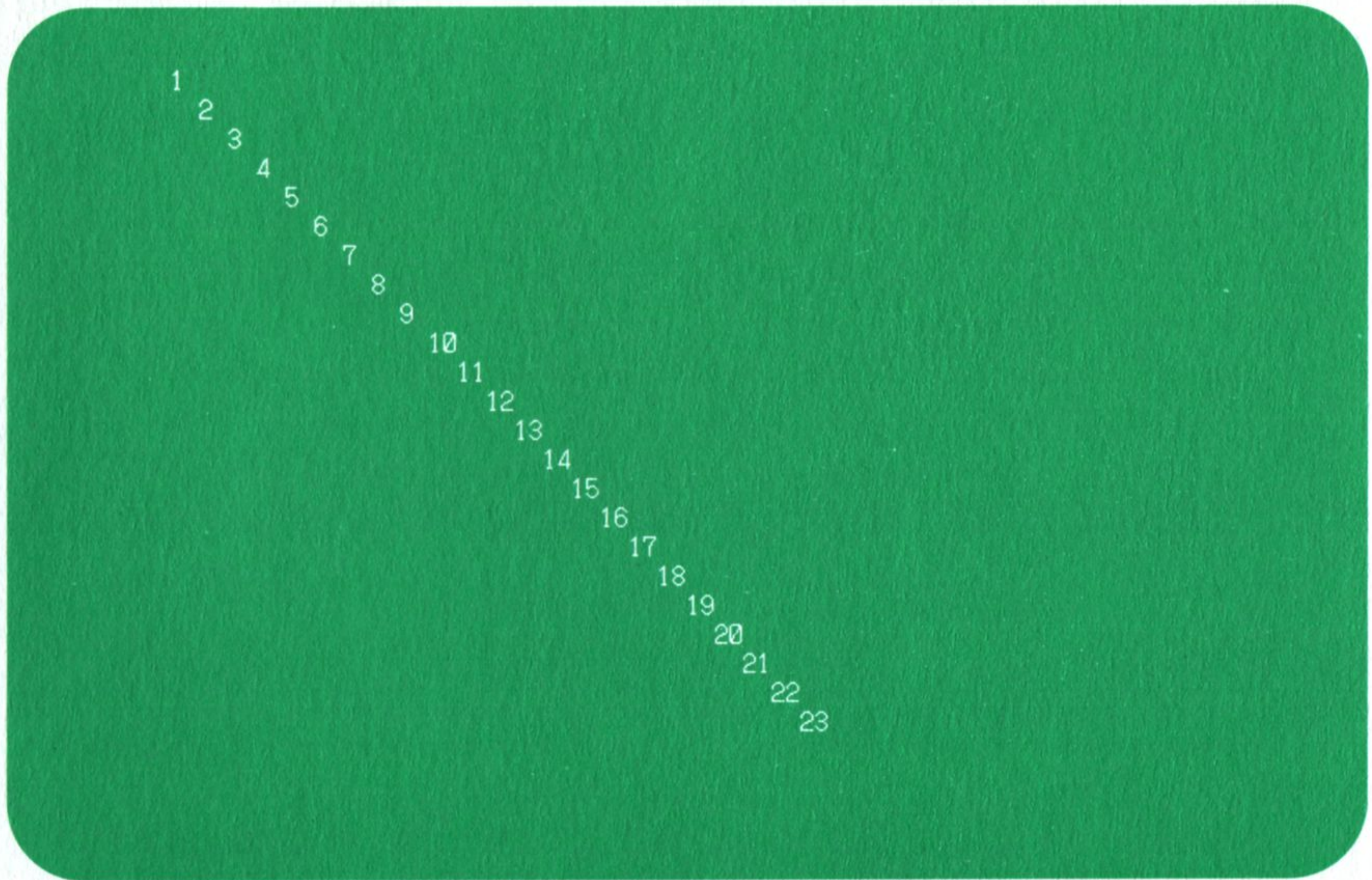
マルチステートメントで上のプログラムを書くと、次のようになりますが、少しみにくいですね。

```
'SPL3-002.BAS
CLS:LOCATE 12, 40: PRINT "Quick":LOCATE 13, 40: PRINT "BASIC":END
```

LOCATE文のYとXに計算式をあてはめることもできます。次のプログラムは、画面上に数字が斜めに表示されるプログラムです。

FOR...NEXTで23回繰り返し、画面下方向に1行ずつ23行までいき、1行ずれるごとに右方向にYの2倍だけ動きます。スタートポイントは、Y=1　X=2です。

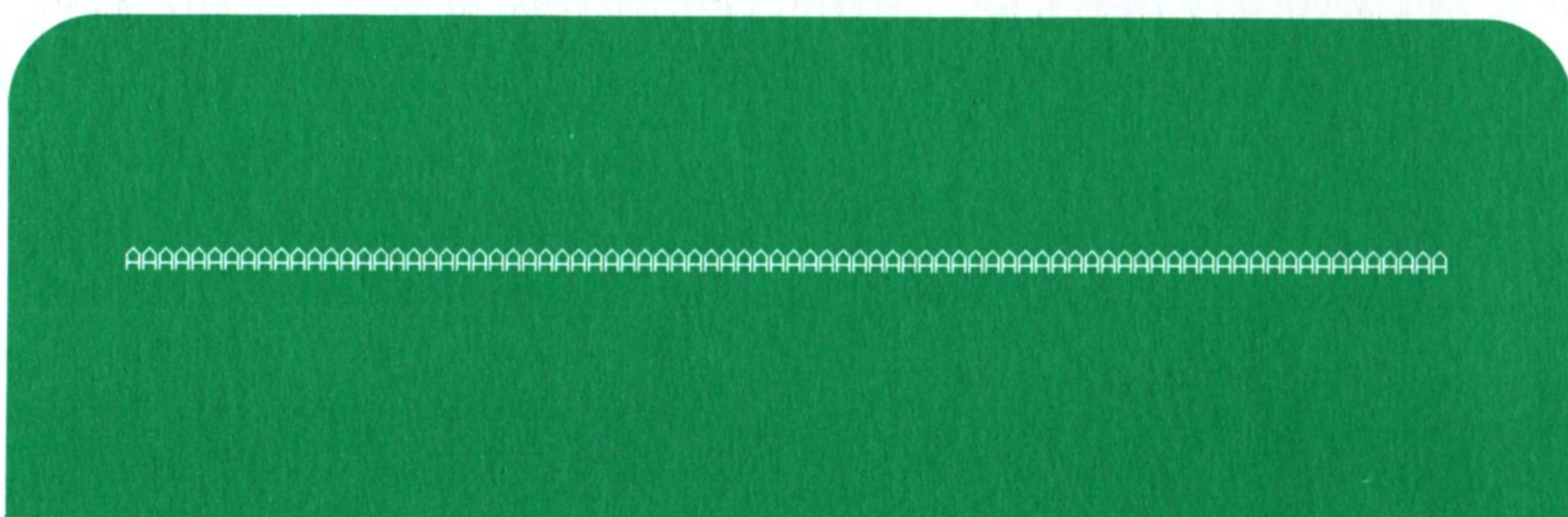
```
'SPL3-003.BAS
CLS
    FOR Y=1 TO 23
       LOCATE Y,Y*2:PRINT Y;   '<---Yの後ろに；（セミコロン）を忘れない
    NEXT                       'ように！
END
```

```
 1
  2
   3
    4
     5
      6
       7
        8
         9
          10
           11
            12
             13
              14
               15
                16
                 17
                  18
                   19
                    20
                     21
                      22
                       23
```

3　文字が動く

PRINT "A"では画面に“A”という文字が表示されました。それでは、PRINT " "ではどうなるでしょう。画面にNULという何もないモノが表示されます。画面上にAを表示して、その上にNULを表示すると画面上のAが消えてしまいます。

そして、次の“A”の表示をLOCATE文でX軸の+の方向に1つずらし、NULで上書きを行なう。これを繰り返すと、画面上をあたかもAという文字が左から右に走っているように見えます。NULを使わなければ、画面1直線に“A”が80個表示されます。次のプログラムを入力してみてください。

```
  F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
                                  SPL3-004.BAS
'SPL3-004.BAS     Aの一直線表示
CLS
    FOR X = 1 TO 80
        LOCATE 10, X: PRINT "A";
            FOR I = 1 TO 200: NEXT
    NEXT

END
```

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

```
  F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
                                  SPL3-005.BAS
'SPL3-005.BAS     Aのランニング
CLS
    FOR X = 1 TO 80
        LOCATE 10, X: PRINT "A";
            FOR I = 1 TO 200: NEXT '時間稼ぎ
        LOCATE 10, X: PRINT " "    'NULの表示
    NEXT

END
```

4　Quick BASICの数値変数について

数値を代入する変数に、“ABC”などという変数名を使います。この変数名は、ある一定の大きさの箱なのです。この数値を入れる箱は、5種類の型に分類されています。実際には、型によって変数が使える数値の桁数が決められています。いったい何桁まで使えるのでしょう。プログラムを作って試してください。

基本的には、4種類の数値型と文字型の5種類があります。

変数名の型	サイズ	扱える値の範囲	解説
単精度実数型	4バイト	－3.402823E+38 ～3.402823E+38	小数点を扱う型で、7桁まで 変数名の形は、AかA!。
倍精度実数型	8バイト	−1.797693134862316D+308 ～1.7976931348623D+308	小数点を扱う型で、15桁まで 変数名の形は、A#。
整数型	2バイト	−32768～32762	小数点以下は四捨五入。形はA%。
倍長整数型	4バイト	−2147483647 ～2147483647	有効桁15桁までの小数点のないもの。変数名の形は、A&。
文字型		最大32767文字まで	可変長と固定長がある。形はA$。

```
'SPL3-006.BAS
'1.単精度実数型
CLS
'A = 1000 + 3
A! = 1000000 / 3
PRINT "単精度実数型 A =  "; A
PRINT "単精度実数型 A!=  "; A!
PRINT

'2.倍精度実数型
A# = 1000000000 / 3
PRINT "倍精度実数型 A#="; A#
PRINT

'3.整数型
A% = 6666 / 100
PRINT "整数型 A%="; A%
PRINT

'4.倍長整数型
A& = 1000000 / 3
PRINT "倍長整数型 A&="; A&
PRINT

END
```

それぞれの変数に入れる式を変えてためしてください。変数A!とAは同じものです。ふたつの変数に違う式を入れて実行すると、あとの式の値が表示されます。

```
単精度実数型 A =    333333.3
単精度実数型 A! =   333333.3

倍精度実数型 A#= 333333333.3333333

整数型 A%= 67

倍長整数型 A&= 333333
```

型の概念

コンピュータでは、データの型というものを重要視していて、厳密に区別しています。それは、機械という制限された器の中でデータを扱うため、人間の頭脳のように、無限のデータを扱うわけにいかないからです。数学的に見ると多少無理がありますが、コンピュータで使う数（すう）の概念はつぎのようになっています。

自然数……物を数えるという素朴な観点から自然発生した数。自然数は、1番目、2番目のように順序を表わしたり、1個、2個というように個数を表わすのに用いる。

整　数……自然数に、0（ゼロ）と-1,-2という数を加えたもの。

実　数……整数に小数点がつくもの。

4.3　数字と文字の桁合わせ

次のプログラムを実行してみてください。

```
PRINT 1
PRINT 10
PRINT 100
PRINT 1000
```

画面上に表示された結果は、次のようになります。

```
1
10
100
1000
```

これらの数字は、次のように揃えることができると、画面がスッキリします。また、3.14などのように小数点以下2桁揃えの表示だと、たいへん品がよくなります。

```
◎  3.14
   3.30
×  3.14159
   3.3
```

1　変数の型で表示が変わる

すでに、小数点以下を四捨五入してしまう整数型（A%とA&）と小数点以下を表示する実数型（A OR A!とA#）について学びました。PRINT USINGを使って桁を揃えた表示を行なうときに、その影響があります。2種類のプログラムを載せていますから、実行して結果を比べてください。

変数への数値の入力が、めんどうくさいので、乱数を発生するコマンドを使いました。乱数を使うときには、最初にRANDOMIZEを設定して、カッコの中に整数を入れて初期設定します。RND(1)は、0.999･･･の数値を発生させますので、本プログラムでは、1000倍して、最大999.999･･･の100の位の数値が作れるように指定してあります。RNDのカッコ内の数値によって同じ値の乱数を作ります。それを避けるためには、RANDOMIZEのカッコの中の数値を変させるのにTIMER関数を使う方法があります。一応決まった形として覚えておくと、なにかと便利です。

```
RANDOMIZE VAL(RIGHT$(TIME$,2))
```

```
'SPL3-007.BAS
CLS
RANDOMIZE (1)             '乱数を発生するための初期値
N = 1                     'カウンター
WHILE N <> 10             'カウント10で繰り返し終了
   A = RND(1) * 1000 / 3  '999.99…の間で乱数を発生それを3で割る
       PRINT USING "####.##"; A  '小数点以下2位までを表示
   N = N + 1                    'カウント
WEND
       PRINT                         '改行
N = 1                        'カウンター（初期値設定）
WHILE N <> 10
     A& = RND(1) * 1000 / 3      '整数型変数名に乱数計算値を入れる
        PRINT USING "No##. "; N;  'カウントナンバーを表示
        PRINT USING "####.##"; A& '下2桁は00になる
     N = N + 1                   'カウント
WEND
END                          'プログラム終了
```

```
  254.96
   35.15
  204.48
  312.59
   35.79
   36.16
  254.27
   88.13
   17.83

 No 1.  247.00
 No 2.  232.00
 No 3.   70.00
 No 4.   74.00
 No 5.  153.00
 No 6.   38.00
 No 7.  163.00
 No 8.   13.00
 No 9.  100.00
```

2　表示位置を指定する

位置の指定は、LOCATE文でした。単純に21行4列の表示を行ないます。最初のプログラムを作って、ホームメニューバーのコピー機能を使って、3回ペーストとしました。変数名をあとから修正します。

```
'SLP3-008.BAS
CLS

N = 1                    'カウンター（初期値設定）
WHILE N <> 22
    A& = RND(1) * 1000 / 3     '整数型変数名に乱数計算値を入れる
      PRINT USING "No##. "; N;  'カウントナンバーを表示
      PRINT USING "####.##"; A& '下2桁は00になる
    N = N + 1                  'カウント
WEND

N = 1                    'カウンター（初期値設定）
WHILE N <> 22
    A& = RND(1) * 1000 / 3     '整数型変数名に乱数計算値を入れる
      LOCATE N, 23 'X軸23から表示
      PRINT USING "####.##"; A& '下2桁は00になる
    N = N + 1                  'カウント
WEND

N = 1                    'カウンター（初期値設定）
WHILE N <> 22
    A& = RND(1) * 1000 / 3     '整数型変数名に乱数計算値を入れる
      LOCATE N, 40 'X軸40から表示
      PRINT USING "####.##"; A& '下2桁は00になる
    N = N + 1                  'カウント
WEND
N = 1                    'カウンター（初期値設定）
WHILE N <> 22
    A& = RND(1) * 1000 / 3     '整数型変数名に乱数計算値を入れる
      LOCATE N, 57 'X軸57から表示
      PRINT USING "####.##"; A& '下2桁は00になる
    N = N + 1                  'カウント
WEND

END                      'プログラム終了
```

```
No 1.  235.00       256.00       225.00       109.00
No 2.  178.00        18.00         5.00       211.00
No 3.  193.00       197.00       192.00        69.00
No 4.   97.00       156.00        33.00        62.00
No 5.  101.00        99.00        34.00       194.00
No 6.  258.00       208.00       266.00        27.00
No 7.    5.00       216.00        95.00       153.00
No 8.  254.00        88.00        15.00       302.00
No 9.  271.00        93.00        99.00        87.00
No10.  236.00       277.00       127.00       262.00
No11.   15.00       275.00       100.00       126.00
No12.  138.00       196.00       316.00        97.00
No13.  288.00       329.00       327.00       306.00
No14.  263.00       304.00       134.00       211.00
No15.  125.00        76.00        93.00       209.00
No16.  321.00       232.00        53.00       143.00
No17.  290.00       327.00        54.00        33.00
No18.   19.00        81.00       216.00       187.00
No19.  317.00       178.00       137.00       231.00
No20.  121.00        35.00       138.00       305.00
No21.  175.00       333.00       238.00       278.00
```

カーソルの移動

Quick BASICでは、カーソル移動キーを押し続けなくても、特殊キーとの組合せで、編集画面上のカーソル移動が行なえます。

1語左	[CTRL]＋[←]か[CTRL]＋[A]	1語左	[CTRL]＋[→]か[CTRL]＋[F]
行頭へ	[CTRL]＋[Q][S]	行末へ	[HELP]か[CTRL]＋[Q][D]
前ページへ	[ROLL DOWN]か[CTRL]＋[R]		
次ページへ	[ROLL UP]か[CTRL]＋[C]		

4.4　文字関数

BASICの面白さは、簡単に絵が描けることと文字列の処理ができるということです。ごく、普通の日本語には、文字列という言葉はありません。文字が集まったものであれば単語になり、単語は集まれば、文あるいは文章となり、文章や図・表などが組み合わさって完成すると文書です。

コンピュータの世界では、データとしてみた場合に、意味不明の数字や文字の羅列があります。このように文章や単語として、意味をなさない文字の集まりを文字

列と理解していいと思います。

1　数値を文字として扱う、その逆

Quick BASICでは、数字は数字、文字は文字とハッキリ区別して使います。そのいい例が、変数名Aの中には、数字が入りません。また、A$の中に入っている数字は、文字として扱われます。

入力した数字のたし算を行なった結果と文字として連結した結果を表示するプログラムです。

```
PRINT 1+2          表示は3    <--------数値として足し算の結果が表示
PRINT "1"+"2"      表示は12   <--------文字が連結されて表示
```

変数名の型にこだわるのは、ここにあるのです。そして、数字を文字に、文字を数字に変えることができる関数が用意されています。数字から文字へはSTR$、文字から数字へはVAL関数を使います。

```
'SPL3-009.BAS
CLS
N = 1
  DIM DAT(5)
    WHILE N <> 6
       INPUT "数字を入力してください= ", A
        D(N) = A
N = N + 1
WEND

B = D(1) + D(2) + D(3) + D(4) + D(5)
    PRINT "入力数値の合計      = "; B
MOJI1$ = STR$(D(1)) + STR$(D(2)) + STR$(D(3))
    PRINT "文字として3つ目まで= "; MOJI1$
MOJI2$ = STR$(D(4)) + STR$(D(5))
    PRINT "文字として残り2つ  = "; MOJI2$
```

```
SUJI1 = VAL(MOJI1$)
SUJI2 = VAL(MOJI2$)
   PRINT "文字のたし算         = "; MOJI1$ + MOJI2$
   PRINT "数値変換後のたし算  = "; SUJI1 + SUJI2
END
```

```
数字を入力してください= 10
数字を入力してください= 20
数字を入力してください= 30
数字を入力してください= 40
数字を入力してください= 50
入力数値の合計      =  150
文字として3つ目まで=  10 20 30
文字として残り2つ  =  40 50
文字のたし算        =  10 20 30 40 50
数値変換後のたし算  =  106080
```

数値を文字として扱うと、数字の前に1文字分の空白が入っています。これは、画面上で、目に見えないだけで、+の記号が入っている状態を表しています。

つまり、文字として"1"+"2"+"3"とすると、画面上では、" 1 2 3"となります。純粋に、1や2だけを文字として扱うには、数字の前の空白を取り去ってやる必要があります。

Quick BASICには、必要な文字だけを抽出するコマンドがたくさん用意してあります。

2　任意の文字を抽出

数字を文字に変換したときの空白や文字列の中から必要な文字のみを抽出するための関数がたくさん用意してあります。とりあえず、よく使う関数を抽出（？）しました。

1．	文字列の右側から1文字抽出	RIGHT$("ABCDE",1)	結果 E
	文字列の右側から2文字抽出	RIGHT$("ABCDE",2)	結果 DE
2．	文字列の左側から1文字抽出	LEFT$("ABCDE",1)	結果 A
	文字列の左側から2文字抽出	LEFT$("ABCDE",2)	結果 AB
3．	文字列の3文字目から2文字抽出	MID$("ABCDE",3,2)	結果 CD
	文字列の2文字目から3文字抽出	MID$("ABCDE",2,3)	結果 BCD

4．ASCIIコードを文字に変換する　CHR$(&H41)　結果 A

5．文字列の中にある文字（列）の位置を知る　INSTER("ABCDE","C")　結果 3

　半角・全角に関係なく文字の位置を知る　KINSTER()

6．文字列の総文字数を知る　LEN("ABCDE")　結果 5

　半角・全角に関係なく文字数を知る　KLEN()

そのほか、次のようなものがあります。

・半角の空白文字列を作る　SPACE$(n)……nは数字

・任意の文字で任意の数だけ文字列を作る　STRING$(n,文字)

・文字列の先頭にある空白を取り除く　LTRIM$()

・文字列の最後にある空白を取り除く　RTRIM$()

・アルファベットの小文字を大文字に変換する　UCASE$(文字列)

・アルファベットの大文字を小文字に変換する　LCASE$(文字列)

次に紹介しているプログラムは、任意に入力された文字列の文字数と文字列の左側から、任意の数の文字列を抽出しました。右側からであれば、RIGHT$に関数名を書き換えると実行できます。さらに、MID$を使って任意の位置から好きな数だけ文字列の抽出を行なっています。

さらに、入力された数字の入力ミスに対するエラー処理がされていますので、参考にしてください。

```
'SPL3-010.BAS
CLS
        INPUT "文字を適当な長さで入力してください= ", MJ$
                PRINT
                PRINT "入力した文字は、「"; MJ$; "」ですね。"
       MJKAZ = LEN(MJ$)       <------入力文字数のチェック
               PRINT "入力したのは、"; MJKAZ; "文字の長さです。"
               PRINT
               PRINT MJKAZ; "以内の数字でお答えください"
   DO
       INPUT "左から何文字までの文字列を抽出しますか=  ", LKAZ
   LOOP UNTIL LKAZ > 0 AND LKAZ < MJKAZ
       LKAZ$ = LEFT$(MJ$, LKAZ)    <-----文字列左からLKAZ個表示
```

```
            PRINT "目的の文字は、「"; LKAZ$; "」です。"
            PRINT
            PRINT "「"; MJ$; "」は"; MJKAZ; "文字の文字列です。"
            PRINT "---------------->特定の文字列を抽出します。"
DO
     INPUT "抽出文字のスタート位置は、何番目ですか=  ", SNO
     IF SNO < 0 THEN
            PRINT "ジョークでしょ?  正のかずを入力してください"
            PRINT
     ELSEIF SNO > MJKAZ THEN
           PRINT "文字列オーバーです。もっと小さなかずを入力してください。"
           PRINT
     END IF
LOOP UNTIL SNO > 0 AND SNO < MJKAZ
     ENO = MJKAZ - SNO
DO
     INPUT "何文字文抽出しますか。", NANO
     IF NANO < 0 THEN
            PRINT "ジョークでしょ?  正のかずを入力してください"
            PRINT
     ELSEIF NANO > ENO THEN
           PRINT "文字列オーバーです。もっと小さなかずを入力してください。"
           PRINT
     END IF
LOOP UNTIL NANO > 0 AND NANO < ENO
     TUMJ$ = MID$(MJ$, SNO, NANO)
            PRINT "目的の文字列は、「"; TUMJ$; "」です。"

END
```

```
文字を適当な長さで入力してください= ABCDEFG

入力した文字は、「ABCDEFG」ですね。
入力したのは、 7 文字の長さです。

 7 以内の数字でお答えください
左から何文字までの文字列を抽出しますか=  3
目的の文字は、「ABC」です。

「ABCDEFG」は 7 文字の文字列です。
---------------->特定の文字列を抽出します。
抽出文字のスタート位置は、何番目ですか=  2
何文字文抽出しますか。2
目的の文字列は、「BC」です。
```

3　入力文字数をカウント

プログラムは、入力時のミスを防止することも大切な考え方の1つです。たとえば、数字の100を入力するところを200と入力したり、3文字制限や4文字制限で入力するような場合に、文字数が多かったり、少なかったりがあります。そんなときに、プログラム上で入力できる文字数を制限してしまうのが、いちばん確実なやり方です。

INKEY$だったらたやすくできますし、文字入力後の[リターン]キーが不要です。

INPUT文では、文字を入力後、[リターン]キーを押してから文字数や文字の種類のチェックを行ないますからワンテンポ遅れます。INPUT文に比べるとプログラムが多少複雑になりますが、入力方法の1つの形として覚えておくと便利です。

ここでは、INKEY$のプログラム例を2つと、ASCIIコード表を表示するプログラムを載せておきます。IF...THENやDO...LOOPを使っていますが、GOTO文がありません。Quick BASICの特長である構造文の原型が含まれています。

1．押されたキーのASCIIコード表示

この関数は、キーボードから1文字を読み込みます。このプログラムを使うと、押したキーのASCIIコードが16進数で表示されます。デタラメにキーを押しているうちに、偶然[E]、[N]、[D]が入力されると、押した順番に関係なくプログラムが終了します。

また、キーの一部が2バイトコードになっています。Quick BASICのときだけかどうか、EXEファイルを作って試してください。

```
'SPL3-011.BAS
  CLS
  DO
    DO
        ANS$ = INKEY$
    LOOP UNTIL ANS$ <> ""
    IF (LEN(ANS$) = 2) THEN
        ASK1$ = HEX$(ASC(LEFT$(ANS$, 1)))
        ASK2$ = HEX$(ASC(RIGHT$(ANS$, 1)))
               PRINT "2バイト16進数  &H"; ASK1$ + ASK2$
    ELSE
        ASK$ = HEX$(ASC(ANS$))
               PRINT "1バイト16進数  &H"; ASK$,
               PRINT "押したキーは、<"; ANS$; ">です。"
    END IF
               PRINT
    IF ANS$ = "E" THEN ANS1$ = "E"
    IF ANS$ = "N" THEN ANS2$ = "N"
    IF ANS$ = "D" THEN ANS3$ = "D"
    TANS$ = ANS1$ + ANS2$ + ANS3$
  LOOP UNTIL TANS$ = "END"
END
```

```
適当なキーを押してみてください
1バイト16進数  &H51          押したキーは、<Q>です。
1バイト16進数  &H75          押したキーは、<u>です。
1バイト16進数  &H69          押したキーは、<i>です。
1バイト16進数  &H63          押したキーは、<c>です。
1バイト16進数  &H6B          押したキーは、<k>です。
1バイト16進数  &H45          押したキーは、<E>です。
1バイト16進数  &H4E          押したキーは、<N>です。
1バイト16進数  &H44          押したキーは、<D>です。
```

２．入力文字の右から左への表示

画面上を入力した文字が右から左に表示されます。文字数をENO=30として、30文字分を表示すれば、終了するようになっています。ENOの数値を変化させると、入力できる文字数が変化します。

日本語入力の場合は、画面下にいったん表示します。[リターン]キーを押すと画面中央に表示されます。変数Nが入力文字数のカウントを行なっています。

```
'SPL3-012.BAS
CLS:N = 0: X = 50: ENO = 30
DO

  DO
     LOCATE 23, 16
          ANS$ = INKEY$
  LOOP UNTIL ANS$ <> ""
     LOCATE 10, X - N
          TTANS$ = TTANS$ + ANS$
               PRINT TTANS$;
          N = N + 1
LOOP UNTIL N > ENO
END
```

左から右に文字表示

Quick BASICを組んで

3．ASCIIコード表作成プログラム

キーボードから入力される文字は、アスキーコードという整数値（10進数で0～127：16進数で&H00～&H7F）に対応してつけられています。コード表は、256個の文字を割り当てることができますから、カタカナの入力できる日本では、アスキーコードにカタカナを加えた拡張コードをJISで設定しています。

ところが、コード表にはまだ余裕があるので、PC-9801では、NECキャラクタがコード表の一部に割り当てられています。拡・拡張アスキーコードあるいは、拡張JISコードということになります。これは、NECだけではなく富士通でも日立でも同様です。そのため、パソコンメーカーによって少しずつ違っている拡張JISコードのことを、「キャラクタコード」という苦しい呼び方をしています。NECの拡張キャラクタはキーボード上からの入力はできません。

1章から使われてきたコマンドだけですから、1つずつプログラムリストを見ていくと理解できると思います。一度に画面に表示できませんので、INKEY$を使って、画面の切り換えを待っています。

```
'SPL3-013.BAS
CLS
 XX = -15: X = 16: EX = 60
 FOR I = &H10 TO &HFF
    YY = (I MOD 16) + 1
    IF XX >= EX AND YY = 1 THEN
        XX = 1
         DO
                ANS$ = INKEY$
                        LOCATE 18, 10
                        PRINT "適当なキーを押してください。"
             IF ANS$ <> "" THEN CLS
        LOOP UNTIL ANS$ <> ""
    ELSEIF YY = 1 THEN
           XX = XX + X
    END IF
                        LOCATE YY, XX
     PRINT USING "&H& & <###> & &"; HEX$(I); I; CHR$(I);
 NEXT I
END
```

&H10	< 16>	DE	&H20	< 32>		&H30	< 48>	0	&H40	< 64>	@	&H50	< 80>	P
&H11	< 17>	D1	&H21	< 33>	!	&H31	< 49>	1	&H41	< 65>	A	&H51	< 81>	Q
&H12	< 18>	D2	&H22	< 34>	"	&H32	< 50>	2	&H42	< 66>	B	&H52	< 82>	R
&H13	< 19>	D3	&H23	< 35>	#	&H33	< 51>	3	&H43	< 67>	C	&H53	< 83>	S
&H14	< 20>	D4	&H24	< 36>	$	&H34	< 52>	4	&H44	< 68>	D	&H54	< 84>	T
&H15	< 21>	NK	&H25	< 37>	%	&H35	< 53>	5	&H45	< 69>	E	&H55	< 85>	U
&H16	< 22>	SN	&H26	< 38>	&	&H36	< 54>	6	&H46	< 70>	F	&H56	< 86>	V
&H17	< 23>	EB	&H27	< 39>	'	&H37	< 55>	7	&H47	< 71>	G	&H57	< 87>	W
&H18	< 24>	CN	&H28	< 40>	(	&H38	< 56>	8	&H48	< 72>	H	&H58	< 88>	X
&H19	< 25>	EM	&H29	< 41>	)	&H39	< 57>	9	&H49	< 73>	I	&H59	< 89>	Y
&H1A	< 26>	SB	&H2A	< 42>	*	&H3A	< 58>	:	&H4A	< 74>	J	&H5A	< 90>	Z
&H1B	< 27>	EC	&H2B	< 43>	+	&H3B	< 59>	;	&H4B	< 75>	K	&H5B	< 91>	[
&H1C	< 28>		&H2C	< 44>	,	&H3C	< 60>	<	&H4C	< 76>	L	&H5C	< 92>	¥
&H1D	< 29>		&H2D	< 45>	-	&H3D	< 61>	=	&H4D	< 77>	M	&H5D	< 93>	]
&H1E	< 30>		&H2E	< 46>	.	&H3E	< 62>	>	&H4E	< 78>	N	&H5E	< 94>	^
&H1F	< 31>		&H2F	< 47>	/	&H3F	< 63>	?	&H4F	< 79>	O	&H5F	< 95>	_

&H60	< 96>	`	&H70	<112>	p	&H80	<128>	▁	&H90	<144>	┴	&HA0	<160>	
&H61	< 97>	a	&H71	<113>	q	&H81	<129>	▂	&H91	<145>	┬	&HA1	<161>	｡
&H62	< 98>	b	&H72	<114>	r	&H82	<130>	▃	&H92	<146>	┤	&HA2	<162>	｢
&H63	< 99>	c	&H73	<115>	s	&H83	<131>	▄	&H93	<147>	├	&HA3	<163>	｣
&H64	<100>	d	&H74	<116>	t	&H84	<132>	▅	&H94	<148>		&HA4	<164>	､
&H65	<101>	e	&H75	<117>	u	&H85	<133>	▆	&H95	<149>	─	&HA5	<165>	･
&H66	<102>	f	&H76	<118>	v	&H86	<134>	▇	&H96	<150>	│	&HA6	<166>	ｦ
&H67	<103>	g	&H77	<119>	w	&H87	<135>	█	&H97	<151>	▕	&HA7	<167>	ｧ
&H68	<104>	h	&H78	<120>	x	&H88	<136>	▏	&H98	<152>	┌	&HA8	<168>	ｨ
&H69	<105>	i	&H79	<121>	y	&H89	<137>	▎	&H99	<153>	┐	&HA9	<169>	ｩ
&H6A	<106>	j	&H7A	<122>	z	&H8A	<138>	▍	&H9A	<154>	└	&HAA	<170>	ｪ
&H6B	<107>	k	&H7B	<123>	{	&H8B	<139>	▌	&H9B	<155>	┘	&HAB	<171>	ｫ
&H6C	<108>	l	&H7C	<124>	¦	&H8C	<140>	▋	&H9C	<156>	╭	&HAC	<172>	ｬ
&H6D	<109>	m	&H7D	<125>	}	&H8D	<141>	▊	&H9D	<157>	╮	&HAD	<173>	ｭ
&H6E	<110>	n	&H7E	<126>	~	&H8E	<142>	▉	&H9E	<158>	╰	&HAE	<174>	ｮ
&H6F	<111>	o	&H7F	<127>		&H8F	<143>	┼	&H9F	<159>	╯	&HAF	<175>	ｯ

&HB0	<176>	ｰ	&HC0	<192>	ﾀ	&HD0	<208>	ﾐ	&HE0	<224>	═	&HF0	<240>	╳
&HB1	<177>	ｱ	&HC1	<193>	ﾁ	&HD1	<209>	ﾑ	&HE1	<225>	╞	&HF1	<241>	円
&HB2	<178>	ｲ	&HC2	<194>	ﾂ	&HD2	<210>	ﾒ	&HE2	<226>	╪	&HF2	<242>	年
&HB3	<179>	ｳ	&HC3	<195>	ﾃ	&HD3	<211>	ﾓ	&HE3	<227>	╡	&HF3	<243>	月
&HB4	<180>	ｴ	&HC4	<196>	ﾄ	&HD4	<212>	ﾔ	&HE4	<228>	◢	&HF4	<244>	日
&HB5	<181>	ｵ	&HC5	<197>	ﾅ	&HD5	<213>	ﾕ	&HE5	<229>	◣	&HF5	<245>	時
&HB6	<182>	ｶ	&HC6	<198>	ﾆ	&HD6	<214>	ﾖ	&HE6	<230>	◥	&HF6	<246>	分
&HB7	<183>	ｷ	&HC7	<199>	ﾇ	&HD7	<215>	ﾗ	&HE7	<231>	◤	&HF7	<247>	秒
&HB8	<184>	ｸ	&HC8	<200>	ﾈ	&HD8	<216>	ﾘ	&HE8	<232>	♠	&HF8	<248>	
&HB9	<185>	ｹ	&HC9	<201>	ﾉ	&HD9	<217>	ﾙ	&HE9	<233>	♥	&HF9	<249>	
&HBA	<186>	ｺ	&HCA	<202>	ﾊ	&HDA	<218>	ﾚ	&HEA	<234>	♦	&HFA	<250>	
&HBB	<187>	ｻ	&HCB	<203>	ﾋ	&HDB	<219>	ﾛ	&HEB	<235>	♣	&HFB	<251>	
&HBC	<188>	ｼ	&HCC	<204>	ﾌ	&HDC	<220>	ﾜ	&HEC	<236>	●	&HFC	<252>	＼
&HBD	<189>	ｽ	&HCD	<205>	ﾍ	&HDD	<221>	ﾝ	&HED	<237>	○	&HFD	<253>	
&HBE	<190>	ｾ	&HCE	<206>	ﾎ	&HDE	<222>	ﾞ	&HEE	<238>	╱	&HFE	<254>	
&HBF	<191>	ｿ	&HCF	<207>	ﾏ	&HDF	<223>	ﾟ	&HEF	<239>	╲	&HFF	<255>	

4　自分のカーソルを作る

プログラムを組むということは、Quick BASIC言語を覚えるということではありません。自分がパソコンにやらせたいことを命令するということです。本書でもここまでにサンプルプログラムを10個ほど紹介していますが、決して、複雑で特殊なBASICは使っていません。さらに、市販のワープロソフトやゲームソフトのように完成されたものもありません。なぜなら、本書で触れているプログラムは、これからプログラムを組む“あなた”へのヒントでしかありません。そのため、あなたが組んでみたいプログラムのヒントが載っていれば、「よい本」になるし、自分の知りたいことへのヒントがなければ「役に立たない本」になるでしょう。

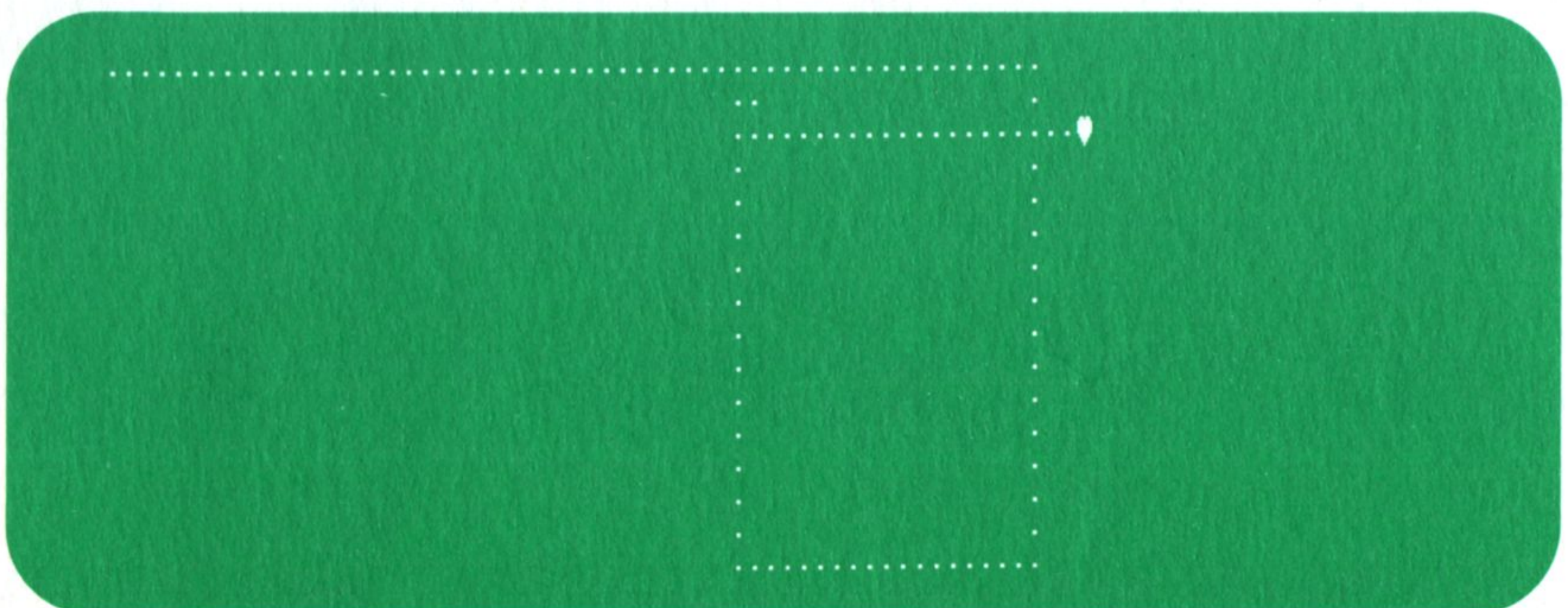
Quick BASICでは、キーボードから入力できないキャラクタをコード表の中からCHR$を使って画面に表示します。好きなキャラクタで自作します。

キーボードのカーソル移動キー、［↑］［↓］［←］［→］を押すとその方向にキャラクタコードから選んだカーソルが移動します。

```
'SPL3-014.BAS

CLS
  KS$ = CHR$(&HE9): YY = 1: XX = 1
DO
    DO
        ANS$ = INKEY$
            GOSUB CSL
    LOOP UNTIL ANS$ <> ""
    IF ANS$ = CHR$(&H0) + CHR$(&H48) THEN
        YY = YY - 1
        IF YY <= 0 THEN YY = 1
            GOSUB CSL

    ELSEIF ANS$ = CHR$(&H0) + CHR$(&H50) THEN
        YY = YY + 1
        IF YY >= 23 THEN YY = 23
            GOSUB CSL
```

```
    ELSEIF ANS$ = CHR$(&H0) + CHR$(&H4B) THEN
        XX = XX - 1
        IF XX <= 0 THEN XX = 1
            GOSUB CSL

    ELSEIF ANS$ = CHR$(&H0) + CHR$(&H4D) THEN
        XX = XX + 1
        IF XX >= 80 THEN XX = 80
            GOSUB CSL
    END IF
LOOP
END

CSL:
  LOCATE YY, XX
        PRINT KS$;
  LOCATE YY, XX
        PRINT " ";
 RETURN
```

プログラムの中のKS$=CHR$(&HE9)をKS$=CHR$(&HE8)、CHR$(&HEA)、CHR$(&HEB)に変更して実行してみてください。

カーソル移動キーのキャラクタコードは、2バイトです。CHR$(&H0)をつけることを忘れないようにしてください。

第5章　ファイル処理の基礎知識

ファイルには、プログラムファイルとデータファイルがあります。ここでは、キーボードから入力したデータをディスクに保存します。

キーボードから入力したデータをディスクに書き込む（なぜか、OUTPUT）、書き込んだディスク内のデータを画面で表示してみる（これがINPUT）。

これらのデータ処理の方法には、シーケンシャルファイル形式とランダムファイル形式の2つをもっています。

簡単にいうと、シーケンシャルファイルは、昔の人が使っていた巻紙という手紙に似ており、ランダムファイルは、レポート用紙のように1枚、1枚管理できるお手紙だと考えていいでしょう。

シーケンシャルファイルは、前から順番にデータを入力して行きますので、必要とするデータを探す場合にも、最初から探し出し、一部分だけを修正しても最初から順番に保存する必要があります。

その点、ランダムファイル形式の場合は、何項目のデータが保存されていても、インデックス（目次）をキッチリ作ってさえあれば、一発で必要な項目だけを引っ張り出して、その項目の部分だけを簡単に修正できます。しかし、データを入れるためには、入力するデータの量や形など、多少複雑な知識が必要となります。このように、Quick BASICのデータファイルの形はこの2種類しかありません。

ランダムファイル形式については、型の指定が面倒なので、本書の入門者向けという主旨からはずれますので割愛します。

5.1　シーケンシャルファイルの作成

ディスクへデータを書き込むためには、

1．キーボードからのデータ入力
2．ディスクのオープン
3．ディスクへのデータの書き込み
4．ディスクのクローズ

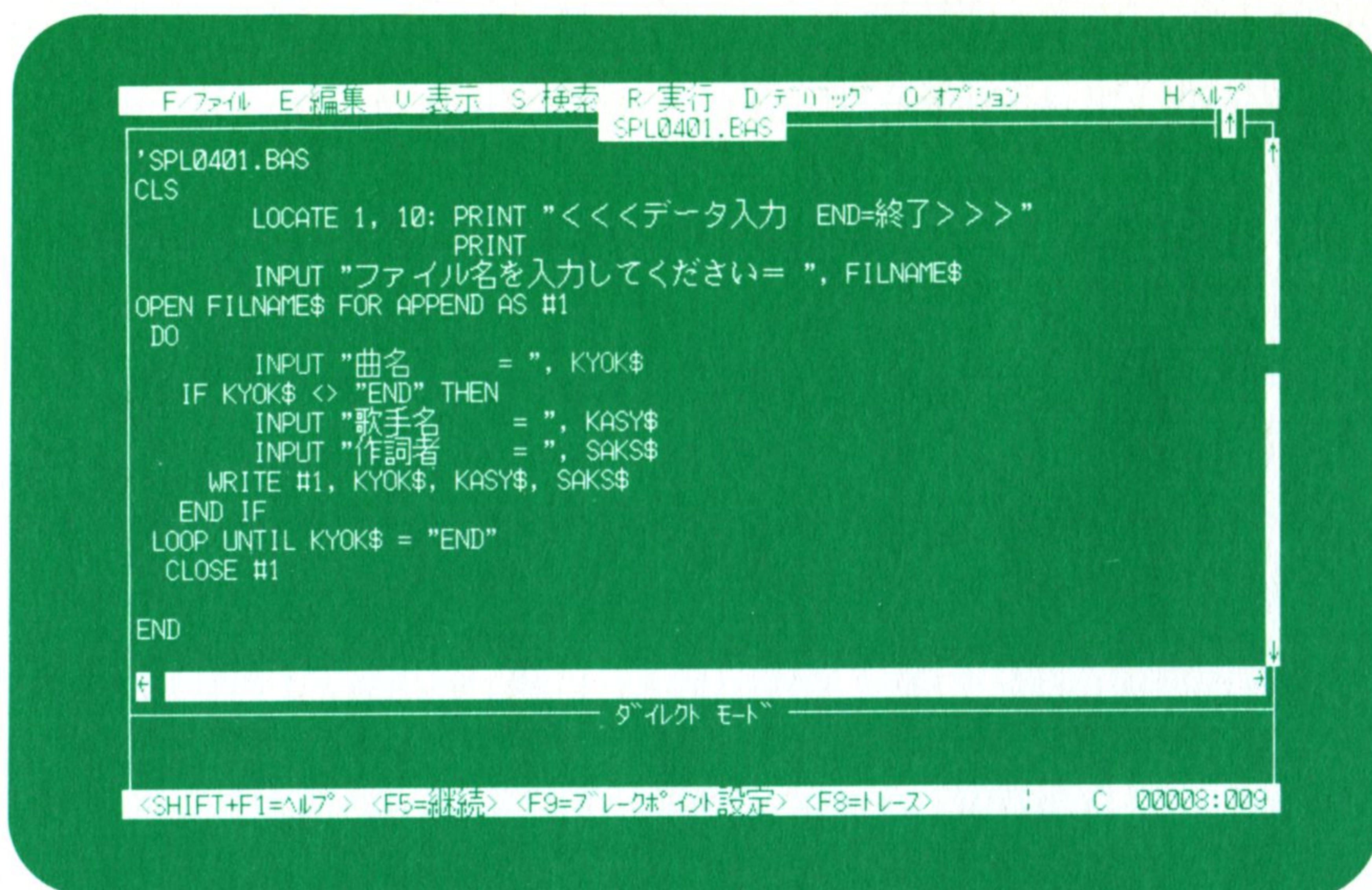
を、行ないます。

また、ディスクに書き込まれているデータを画面上に表示させるには、

1．ディスクのオープン
2．画面上への表示
3．ディスクのクローズ

の操作を行ないます。

1　入力データをディスクへ読み込む

キーボードからのデータ入力は、ENDを入力すると終了します。DO...LOOPの中の項目は、INPUT文を使って、自由に増やすことができます。その場合には、WRITE #1のあとの変数名を、INPUTする変数名と合わせる必要があります。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
                                SPL0401.BAS
'SPL0401.BAS
CLS
        LOCATE 1, 10: PRINT "<<<データ入力  END=終了>>>"
                      PRINT
        INPUT "ファイル名を入力してください= ", FILNAME$
OPEN FILNAME$ FOR APPEND AS #1
 DO
        INPUT "曲名      = ", KYOK$
   IF KYOK$ <> "END" THEN
        INPUT "歌手名    = ", KASY$
        INPUT "作詞者    = ", SAKS$
     WRITE #1, KYOK$, KASY$, SAKS$
   END IF
 LOOP UNTIL KYOK$ = "END"
  CLOSE #1

END
                              ダイレクト モード
<SHIFT+F1=ヘルプ> <F5=継続> <F9=ブレークポイント設定> <F8=トレース>   C  00008:009
```

```
          <<<データ入力  END=終了>>>
曲名      = 魔女達（ウィチィズ）
歌手名     = 中山  美穂
作詞者     = 康珍化
曲名      = 素直にI'm Sory
歌手名     = チェッカーズ
作詞者     = 藤井郁弥
曲名      = 君の弱さ
歌手名     = 渡辺  美里
作詞者     = 渡辺  美里
曲名      = たとえばフォーエバー
歌手名     = 小泉今日子&真田弘之
作詞者     = 和田  誠
曲名      = ガラスの目隠し
歌手名     = 小川  範子
作詞者     = 川村  真澄
曲名      =
```

2　ディスクから画面にデータを表示

プログラム中のEOFは、「END OF FILE（ファイルのおしまい）」ということで、保存されているデータをすべて表示させるのに必要な命令です。詳しくは、マニュアルを見るとしても、ここでは、この形式を丸ごと覚えてください。

INPUT #1のあとの変数名は、キーボードから入力したときの変数名の必要はありませんが、文字変数と数値変数の型は合わせる必要があります。

こうして、ディスクからデータが読み出されることが、確認できたら、ディスクをクローズしたあと、検索や並べ替えのプログラムを追加していきます。　データの細工や画面の体裁を自分用に作り変えていきます。

```
'SPL0402.BAS
 CLS
      LOCATE 1, 10: PRINT "ディスクからのデータ読み込み"
                    PRINT
        INPUT "ファイル名を入力してください= ",FILNAME$
OPEN FILNAME$ FOR INPUT AS #1
 DO UNTIL EOF(1)
        INPUT #1, KYOK$, KASY$, SAKS$
  PRINT KYOK$, KASY$, SAKS$
 LOOP
  CLOSE #1
 PRINT "表示終了！　適当なキーを押してください"
 DO
 ANS$ = INKEY$
 LOOP UNTIL ANS$ <> ""

END
```

```
       ディスクからのデータ読み込み
魔女達（ウィチィズ）          中山　美穂    康珍化
素直にI'm Sory              チェッカーズ  藤井郁弥
君の弱さ      渡辺　美里    渡辺　美里
たとえばフォーエバー        小泉今日子&真田弘之        和田　誠
ガラスの目隠し              小川　範子    川村　真澄
```

5.2　ロータス1-2-3用データ入力プログラム

シーケンシャルファイル形式で、「ロータス1-2-3」で使えるデータ入力プログラムを作ります。このプログラムでは、自分で任意の件数だけ項目を指定して、自分で項目名を入力します。前述の歌手名登録プログラムに比べると大きくなっていますが、シーケンシャルファイル形式ですから原則は変わりません。

プログラムを順番に一部分ずつ抽出しながら説明しますが、全プログラムリストは、122ページに掲載しています。

1　データ入力のメニュー

Quick BASICの場合は、繰り返しを行なうコマンドの中で比較的扱いやすいのは、DO...LOOPです（単に、個人の趣味だったりして……）。

プログラムを組む上での使い方は、何度か触れてきましたが、

```
DO
.....
.....
LOOP UNTIL～
```

と、なっています。DOのあとは、LOOPだけでもいいのですが、LOOPの中から抜け出せなくなるので、UNTILやWENDを使ってLOOPからの脱出を行ないます。

また、マニュアルや多くの解説書に書いてあるブロック構造とは、DO...LOOPに囲まれたプログラムのことです。

この繰り返しの原型になるのが次のメニュープログラムです。

プログラムを順番に見てみます。

まず、繰り返しを行なうためのDOがあり、続いて画面をクリアするためのCLSが書いてあります。非常に単純です。続いて、メニューのメッセージを表示する位置を決めて、PRINT文を繰り返しています。LOCATE文は、文字列を画面に表示する

```
DO
CLS
    LOCATE 1, 10: PRINT "シーケンシャルデータの読み書き"
    LOCATE 3, 12: PRINT "データ入力          <1>"
    LOCATE 4, 12: PRINT "ディスクの読みだし<2>"
    LOCATE 5, 12: PRINT "終了              <3>"
    LOCATE 7, 10: INPUT "番号を入力してください=   ", FL
   IF FL = 5 THEN END
LOOP UNTIL FL > 0 AND FL < 4
```

ためのものです。Quick BASICでは、N88-BASICとXとY位置の指定が反対です。

メニューの項目名表示のあとで、IF...THENの判断で、変数FLが5だったらプログラムを終了させます。

そして、FLが0より小さい（正確にいうと、FLが0より大きくなかったらとなります）か、FLが4より大きいとき(同様にFLが4より小さくなかったらが解釈は正しい)は繰り返しを行ないます。繰り返しを行なうたびに、DOのすぐあとにCLSがあるので画面がいったんクリアされてからメニューが表示されます。

DOとCLSの入力位置を変えて、違いを試してください。

```
シーケンシャルデータの読み書き

  データ入力        <1>
  ディスクの読みだし<2>
  終了              <3>

番号を入力してください=
```

2　ドライブの指定

ドライブの指定は、N88-DISK BASICの場合は、ドライブ1、ドライブ2ですが、MS-DOSの場合は、ドライブをA、B、Cと呼びます。

パソコンにハードディスクやRAMディスクを使っていると、ドライブの指定が変わります。サンプルプログラムでは、ドライブの制限をA、B、Cとしています。必

要に応じて、LOOP UNTILのあとにOR DRV$="D"・・・・を書き加えてください。このとき、ドライブAを指定するつもりで入力したところ、ASSSなどと入力しても、文字列の左側の1文字だけを読むように、文字関数のLEFT$が使われています。

また、A、B、C以外の指定であったら再びメニューが表示されます。DO...LOOPで繰り返すようにしているのは、メニューのときと同じです。

```
DO
  INPUT "ドライブを指定してください(A.B...)=  ", DR$
        DRV$ = LEFT$(DR$, 1)
        DR$ = ""
LOOP UNTIL DRV$ = "A" OR DRV$ = "B" OR DRV$ = "C"
```

3　ファイル名の入力

「ロータス1-2-3」で、使いやすいように拡張子の“.PRN”は、プログラム内で自動的に処理するようにします。そこで、拡張子の“.”（ピリオド）が、ファイル名の中に入らないように文字関数INSTRを使ってチェックしています。チェックに引っかかったときは再入力ができるように、ファイル名入力でもDO...LOOPを使っています。IFの判断として、入力したファイル名の中にカンマが入っていると文字列のどの位置であるかを数字で返してきます。カンマがなければ、0の数字が返されます。それが最初のIFです。

さらに、ELSEIFと続けてファイル名の入力が8文字を越えても、LENで入力字数をチェックして8文字を越えたときには、LEFT$で9文字目以降を切り捨てています。ドライブ名とファイル名をFILNAME$にまとめて、サブルーチンに飛ばしています。サブルーチンの名前はAPPENとOPUTです。

```
DO
  INPUT "ファイル名を入力してください(8文字まで判読)=", FA$
  IF INSTR(FA$, ".") <> 0 THEN
                  PRINT "      カンマをつけないで・・・・!"
        FA$ = ""
  ELSEIF LEN(FA$) > 8 THEN
```

```
          FA$ = LEFT$(FA$, 8)
    END IF
  LOOP UNTIL FA$ <> ""
                PRINT DRV$; ":"; FA$; ".PRN";
                PRINT
    FILNAME$ = DRV$ + ":" + FA$ + ".DAT"
```

4　サブルーチン・プロシージャ

Quick BASICは、構造化プログラミングを行なうため、プロシージャ（手続き）という技法が使えます。プロシージャにしても、たくさんの方法がありますが、独断と偏見をもって、とりあえず、サブルーチン・プロシージャについて触れていきます。

そのほかのものについては、徐々にやっていけばよいでしょう（本書では説明を省きます）。

従来のBASICは、GOSUB...RETUENで、図Aのような形式で書きました。

Quick BASICでは、サブルーチンプロシージャを指定してやると、サブルーチンプログラムを作る画面を自動的に切り換えてくれます。そのため、メインプログラムに惑わされることなく、サブルーチンプログラム制作に専念できます(図B)。

この章では、メニュー表示をメインプログラムとして、キーボードからのデータ入力をサブルーチン1、ディスクの中のデータを画面に表示させるプログラムをサブルーチン2として、プロシージャの指定を行ないました。

簡単にいうと、プログラムのスタートのときに、サブルーチンとして使うプログラム名と、そのサブルーチンプログラムの中で使う変数名を指定してやることが、サブルーチンプロシージャの指定です。

プロシージャを使うと自動的にSUBルーチンの画面が切り換わります。自分の意志で画面切り換えを行なう場合はメインメニューバーの[V/表示]か[SIFT]＋[f･2]で画面切り換えができます。

図A

メニュープログラム

メインプログラム

GOSUBU

RETURN

RETURN

GOSUBU

サブプログラム１

サブプログラム２

図B

プロシージャ指定1

プロシージャ指定2

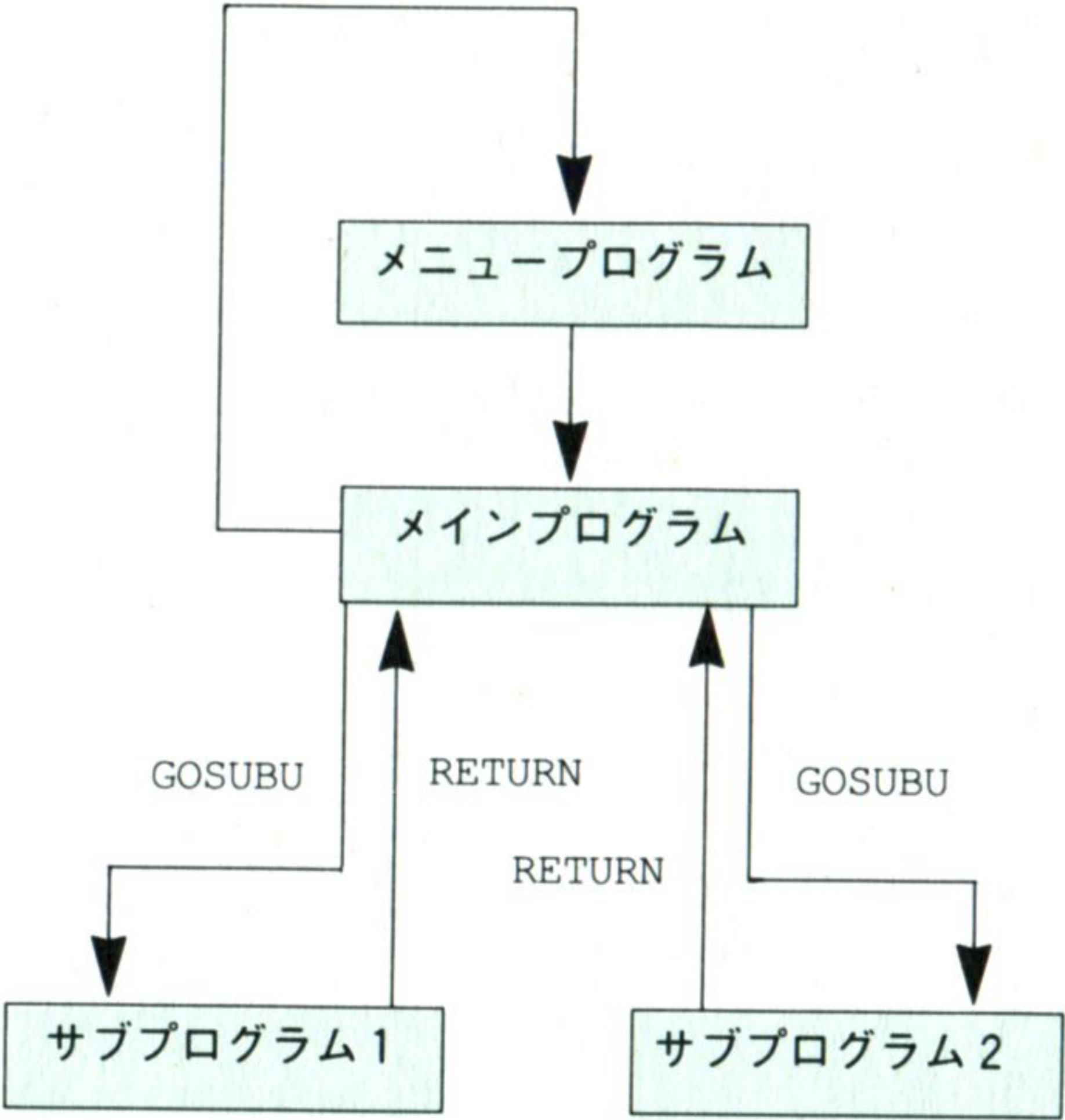
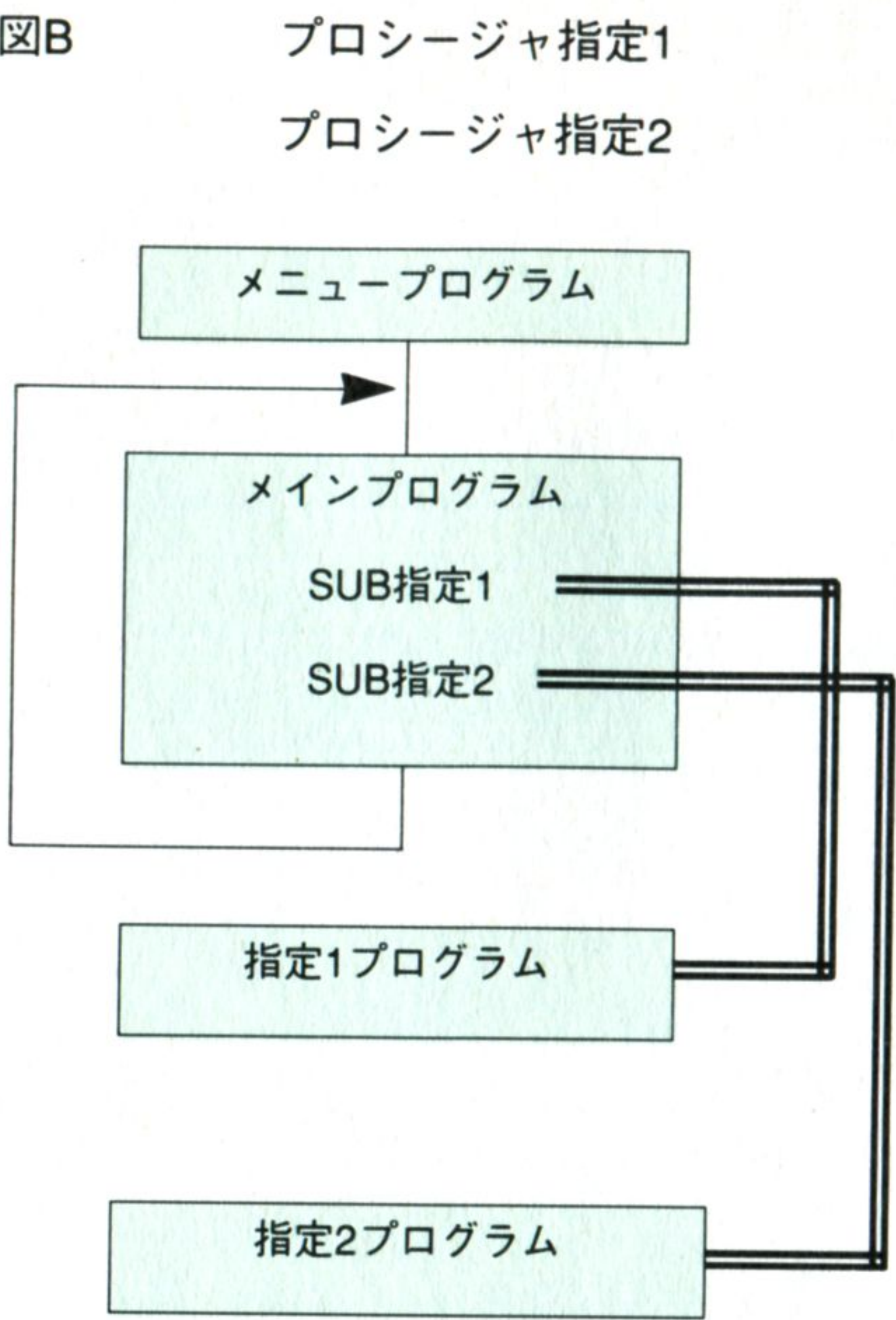
メニュープログラム

メインプログラム

SUB指定1

SUB指定2

指定1プログラム

指定2プログラム

```
DECLARE SUB APPN (FILNAME$)  <-----サブルーチン1の指定
DECLARE SUB OPUT (FILNAME$)  <-----サブルーチン2の指定
          .....
     .....
IF FL = 1 THEN
   APPN (FILNAME$)          <-----サブルーチン1の呼び出し
ELSEIF FL = 2 THEN
       OPUT (FILNAME$)       <-----サブルーチン2の呼び出し
END IF
```

5 サブルーチンプロシージャの削除

サブルーチンプログラムは、サブルーチンプロシージャを設定すると、Quick BASICの画面が自動的に別の画面に換わりますが、必要ないサブルーチンの場合は、メインメニューバーの[V/表示]か[f.2]で次の画面を表示して、サブルーチンプログラム名を削除します。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
'SPL0403.BAS ***  S/SUB一覧...          F2  *****************
'             *   U/実行画面             F4  ム              *
'             *** L/インクルードファイル表示   *****************
'サブ・プロシジャ -----------------------------------------------
DECLARE SUB APPN (
DECLARE SUB IPPN (FILNAME$)

DIM KOMOK$(35), INDAT$(35): NON = 0: YAN = 1
'エラー処理 ---------------------------------------------------
ON ERROR GOTO ERSYOL
ERSYOL:
```

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション          H/ヘルプ
 ----------------------------- S/SUB一覧 -----------------------------
  C/編集するプロシージャ・ウィンドウを選択してください

   SPL0403.BAS
     APPN
     IPPN

  SPL0403.BAS:=メインモジュール

  < A/現在のウィンドウで編集 >

                              < D/削除 >  < 取消 >  < H/ヘルプ >
F1=ヘルプ  RETURN=実行  ESC=取消  TAB=次項目  カーソルキー=選択    :    C  00037:003
```

⑥ ロータス用データ入力全プログラムリスト

すでに保存されているデータファイル名（拡張子.PRNのみ）の表示一覧とエラー処理プログラムが追加されています。画面の体裁を整えるため、第2部で触れるグラフィックスとカラーについても触れています。特に理解できないBASIC言語はないはずです。

```
'SPL0403.BAS *****************************************************
'            *『ロータス1-2-3』用データ入力プログラム              *
'            *****************************************************
'サブ・プロジャの宣言-------------------------------------------------
DECLARE SUB APPN (FILNAME$)
DECLARE SUB IPPN (FILNAME$)

DIM KOMOK$(35), INDAT$(35): NON = 0: YAN = 1
'エラー処理 ---------------------------------------------------------
ON ERROR GOTO ERSYOL
ERSYOL:
    IF ERR = 53 THEN
            SK = 1
            RESUME NEXT
        ON ERROR GOTO 0
    ELSEIF ERR = 71 THEN
                PRINT DRV$; ":¥ "
                PRINT "ドライブの準備ができていません"
                PRINT "準備が終わったら適当なキーを押してください"
  COLOR 6
   PRINT "------------------------------------------------------------"
  COLOR 7
        DO
            ANS$ = INKEY$
        LOOP UNTIL ANS$ <> ""
                RESUME 0
```

```
      ON ERROR GOTO 0
    END IF
'プログラムのスタート------------------------------------------------
CLS
    LINE (0, 15)-(640, 32), 1, BF: LINE (34 * 8, 16)-(60 * 8, 32), 7, BF
LOCATE 2, 20: PRINT "★データ入力  "; : COLOR 0: PRINT "  拡張子は自動的に.PRNです"
     COLOR 7: PRINT
DO
   INPUT "ドライブを指定してください(A.B...)=  "; DR$
       DRV$ = LEFT$(DR$, 1)
       DR$ = ""
LOOP UNTIL DRV$ = "A" OR DRV$ = "B" OR DRV$ = "C"
   COLOR 6
    PRINT "----------------------------------------------------------"
   COLOR 7
    FILES DRV$ + ":" + "*.PRN"
   COLOR 6
    PRINT "----------------------------------------------------------"
   COLOR 7
     INPUT "ファイル名を入力してください(8文字まで判読)=", FA$
DO
   IF INSTR(FA$, ".") <> 0 THEN
                PRINT "      カンマをつけないで・・・・!"
        FA$ = ""
   ELSEIF LEN(FA$) > 8 THEN
        FA$ = LEFT$(FA$, 8)
  END IF
LOOP UNTIL FA$ <> ""
        FILNAME$ = DRV$ + ":" + FA$ + ".PRN"
            CLOSE #1
        OPEN FILNAME$ FOR INPUT AS #1
     IF SK <> 1 THEN
        DO UNTIL EOF(1)
```

```
              INPUT #1, D1$
          LOOP
        SNK = 0
   ELSEIF SK = 1 THEN
    SNK = 1
                  CLOSE #1
  END IF
                  CLOSE #1
  DO
     IF SNK = 0 THEN
          INPUT "データを追加しますか(T=はい/N=新規/E=終了)"; ANS$
          IF ANS$ = "T" THEN ANS$ = "": APPN (FILNAME$)
          IF ANS$ = "N" THEN ANS$ = "": IPPN (FILNAME$)
          IF ANS$ = "E" THEN EXIT DO
    ELSEIF SNK = 1 OR ANS$ = "N" THEN
          SNK = 0
          IPPN (FILNAME$)
     END IF
  LOOP UNTIL ANS$ = "T" AND ANS$ = "N"
CLS
LOCATE 10, 20: PRINT "ｵｼﾏｲ! ｵﾂｶﾚ ｻﾏ ﾃﾞｼﾀ!"
END

    <-----------ここにSUB APPN(FILNAME$)と入力すると自動的に画面が切り
                換わります。続けて、125ページのプログラムを入力します。
    <-----------ここにSUB IPPN(FILNAME$)と入力すると自動的に画面が切り
                換わります。続けて、126ページのプログラムを入力します。
```

●追加データ入力のサブ・プロシージャ

データの新規・追加入力のサブルーチンプログラムです。Quick BASICでは、メインメニューバーの[V/表示]からサブメニューを使うか、[f･2]で画面が切り換えられます。

```
SUB APPN (FILNAME$)
   N = 0: ENF = 0: KENN = 1
DIM KOMOK$(35), INDAT$(35): N = 0: Q$ = CHR$(34): R$ = CHR$(44)
  CLS
 LINE (0, 15)-(640, 32), 1, BF: LINE (34 * 8, 16)-(60 * 8, 32), 7, BF
LOCATE 2, 20: PRINT "◎データ追加  "; : COLOR 0: PRINT "  拡張子は自動的に.PRNです";
   COLOR 7: PRINT FILNAME$
   PRINT
  CLOSE #1
      OPEN FILNAME$ FOR INPUT AS #1
DO
      INPUT #1, DAT$
           N = N + 1
      IF N = 1 THEN KOMOK$(N - 1) = DAT$:
      IF N <= VAL(KOMOK$(0)) + 1 THEN KOMOK$(N - 1) = DAT$
LOOP UNTIL EOF(1)
  CLOSE #1
      PRINT "項目数は、"; : COLOR 4: PRINT KOMOK$(0); : COLOR 7: PRINT "です"
      EDT = VAL(KOMOK$(0))
      KENN = N \ EDT
  OPEN FILNAME$ FOR APPEND AS #1
        PRINT "『◎データを追加します (END=E//  BACK=B//) ◎』"
      KO$ = KOMOK$(EDT): RC = INSTR(KO$, CHR$(13))
      IF RC <> 0 THEN KOMOK$(EDAT) = LEFT$(KO$, RC - 1)
DO
      PRINT "検体番号=  "; : COLOR 6: PRINT KENN; : COLOR 7: PRINT "番"
 FOR I = 1 TO EDT
   PRINT I; ") ";
    COLOR I MOD 6 + 1
        PRINT KOMOK$(I);
        COLOR 7: INPUT "= ", INDAT$(I)
      IF INDAT$(I) = "B//" THEN I = I - 2
      IF I <= 0 THEN I = 0
      IF INDAT$(1) = "E//" THEN I = EDT: ENF = 1
```

```
        IF I = EDT AND ENF <> 1 THEN
      FOR J = 1 TO EDT
   IF J = EDT THEN PRINT #1, Q$; INDAT$(J); Q$ ELSE PRINT #1, Q$; INDAT$(J); Q$; R$;
      NEXT
    END IF
   NEXT
        KENN = KENN + 1
            PRINT
 LOOP UNTIL INDAT$(1) = "E//"
        CLOSE #1
END SUB
```

●新規データ入力のサブ・ルーチンプロシージャ

新規データの入力を行ないます。データ入力メインプログラムでサブルーチンプロシージャーを指定すると、画面が自動的に新規データ入力のサブルーチンプログラムになります。画面が切り換わったとき、SUB IPPN (FILNAME$)とEND SUBの2行だけプログラムが自動的に作成されます。この2行の間にサブ・ルーチンプログラムを作ります。メインプログラムやほかのサブルーチンプロシージャーの画面は、メインメニューバーの[V/表示]を選択してから、サブメニューを使うか、[f･2]で画面が切り換えます。

```
 SUB IPPN (FILNAME$)
 DIM KOMOK$(35), INDAT$(35): KENN = 1: N = 0
 Q$ = CHR$(34): R$ = CHR$(44)
 ENF = 0
 CLS
     LINE (0, 15)-(640, 32), 1, BF: LINE (38 * 8, 16)-(64 * 8, 32), 7, BF
 LOCATE 2, 20: PRINT "◇新規データ入力  "; : COLOR 0: PRINT "  拡張子は自動的に.PRNです";
      COLOR 7: PRINT FILNAME$
      PRINT
 CLOSE #1
```

```
 DO
 INPUT "項目数をいれてください（30項目まで:不明の時は、99）＝  ", KOM$
   KOM = VAL(KOM$)
      IF KOM = 99 THEN PRINT "最後に  //  を入力してください": KOM = 30
       IF KOM = 0 THEN KOM = 99
 LOOP UNTIL 30 >= KOM
DO
 N = N + 1: COLOR N MOD 6 + 1
   PRINT N;
   COLOR 7: INPUT "番目の項目は＝  "; KOMOK$(N)
LOOP UNTIL KOMOK$(N) = "//" OR KOM = N OR N = 30
       IF KOMOK$(N) = "//" THEN KOMOK$(N) = "": N = N - 1
           KOMOK$(0) = STR$(N)
       OPEN FILNAME$ FOR OUTPUT AS #1
   FOR I = 0 TO N
     INDAT$(I) = KOMOK$(I)
       IF I = 0 THEN
           PRINT #1, Q$; INDAT$(0); Q$
       ELSEIF I = N THEN
           PRINT #1, Q$; INDAT$(I); Q$
       ELSE
           PRINT #1, Q$; INDAT$(I); Q$; R$;
     END IF
  NEXT
 CLOSE #1
   NE = N
       PRINT
           OPEN FILNAME$ FOR APPEND AS #1
       PRINT:PRINT "『◇ データ入力を行います（END=E// BACK=B//）◇』"
DO
     PRINT "検体番号＝  "; : COLOR 6: PRINT KENN; : COLOR 7: PRINT "番"
  FOR I = 1 TO NE
       PRINT I; ") ";
```

```
            COLOR I MOD 6 + 1
            PRINT KOMOK$(I);
            COLOR 7: INPUT "= ", INDAT$(I)
        IF INDAT$(I) = "B//" THEN I = I - 2
        IF I <= 0 THEN I = 0
        IF INDAT$(1) = "E//" THEN I = NE: ENF = 1
        IF I = NE AND ENF <> 1 THEN
    FOR L = 1 TO NE
    IF L = NE THEN PRINT #1, Q$; INDAT$(L); Q$ ELSE PRINT #1, Q$; INDAT$(L); Q$; R$;
    NEXT
        END IF
 NEXT
            KENN = KENN + 1:PRINT
        LOOP UNTIL INDAT$(1) = "E//"
  CLOSE #1
END SUB
```

シーケンシャルでデータを保存する場合、WRITE#1を使った文字変数は、""に囲まれ、変数名ごとに,（カンマ）で区切られます。数値変数を使うと、""に囲まれませんから、数値もいったんは、文字変数にしまいます。

```
WRITE #1, KYOK$, KASY$, SAKS$
```

もし、PRINT#1であれば、前処理として、DB$=CHR$(34):KA$=CHR$(44)を作って、

```
PRINT #1,DB$;KYOK$;DB$;KA$;DB$;KASY$;DB$;KA$;DB$;SAKS$;DB$
```

と、書き直してやる必要があります。

ちなみに、CHR$(34)は、"（ダブルクォーテーション）。CHR$(44)は、,（カンマ）を示しています。キャラクタ表を参照してください。

```
"00004","04","","","","","","","","","","","",""
"00005","05","404203","7060","32496","2922","46219","7060","65988","0","80593","
0","211403","0"
"00006","06","216127","","38420","","49400","","56356","","55620","","54751",""
"00007","07","388731","0","23039","0","35693","0","60651","0","65981","0","22640
6","0"
"00008","08","714185","4709","84213","491","174653","4709","106402","","122686",
"","310444",""
"00009","09","275770","","44510","","62831","","59994","","52004","","100603",""
"00010","10","489825","0","43106","0","58893","0","70180","0","88827","0","27192
5","0"
"00011","11","228768","147617","284775","13194","555399","130180","364856","1111
7","328419","5356","1038794","964"
"00012","12","1332300","","154987","","289350","","197525","","216788","","62863
7",""
"00013","13","832190","266467","147094","59370","323555","194124","129355","2859
7","71795","3212","307485","40534"
"00014","14","1022141","27654","151618","466","262364","20431","197061","3597","
156290","2626","406426","1000"
"00015","15","","","","","","","","","","","",""
"00016","16","519580","2173","56854","","89508","2173","89287","","100357","","2
38523",""
```

▲保存されているデータの形

●その他の入力データ用サブルーチンプロシージャ

１．入力データ画面表示サブルーチンプロシージャや、２．印字出力用サブ・ルーチンプロシージャーなどが考えられます。データ入力プログラムを参考に、オリジナルのプログラム作成にチャレンジしてください。並べ替えや合計などの計算式のサブルーチンプロシージャも考えられます。しかし、これらは、「ロータス1-2-3」に読み込みさえすれば、「ロータス1-2-3」上で処理が可能です。

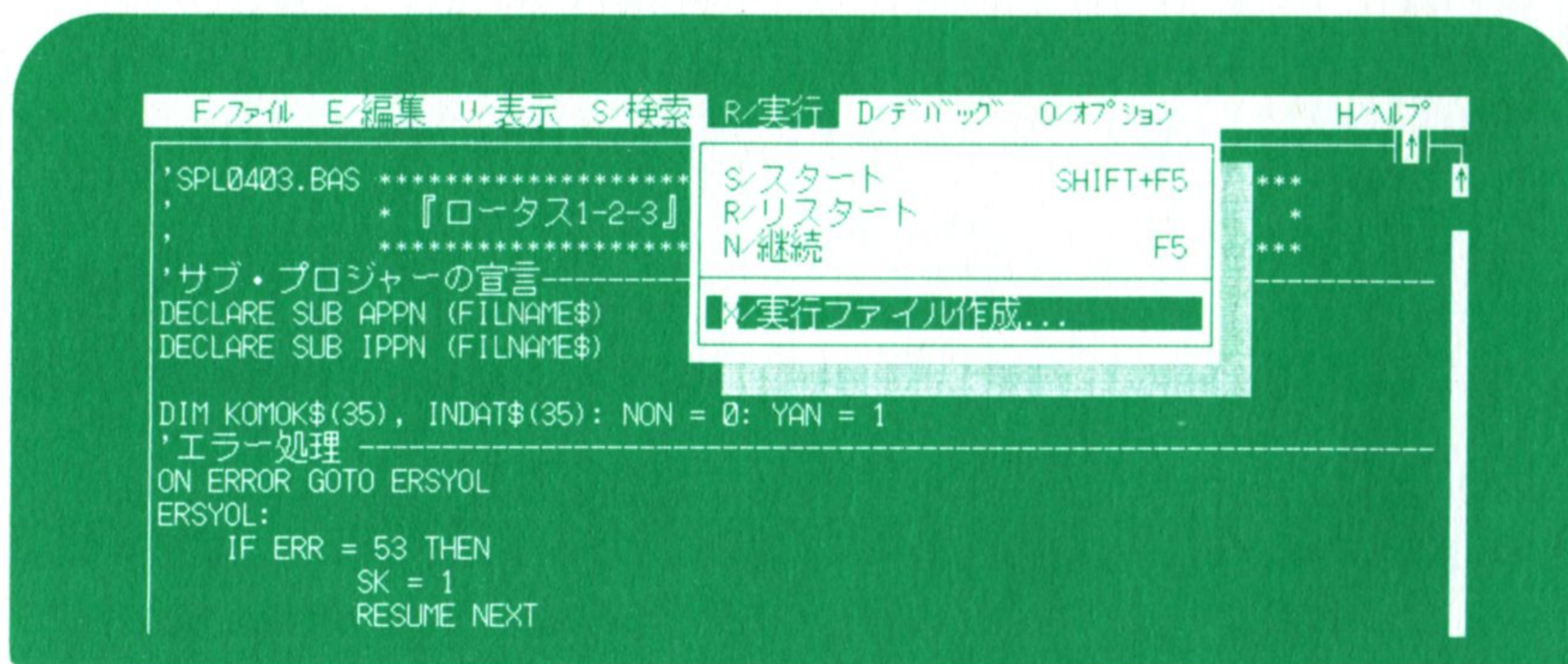
5.3　データ入力プログラムの実行可能(.EXE)ファイルの作成

プログラムの開発には、対話式のインタプリタが便利ですが、いったん、完成したプログラムは、このままでは、プログラムを実行するたびにQuick BASICを起動しなければならないという不便さがあります。

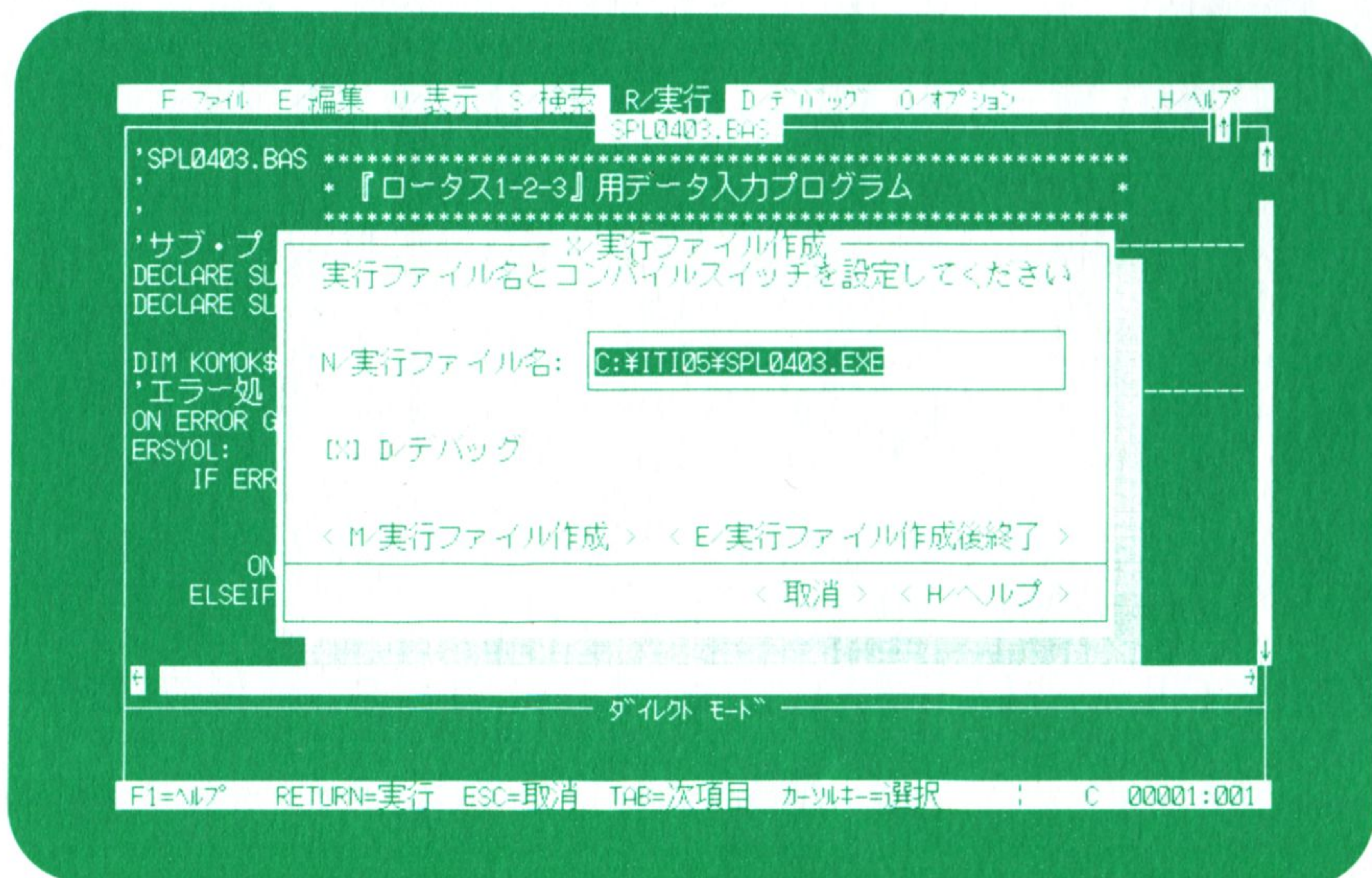
実行可能ファイルにコンパイルすると、MS-DOSのA>が表示されている画面から実行が可能になります。

1　EXEファイル作成のメニュー選択

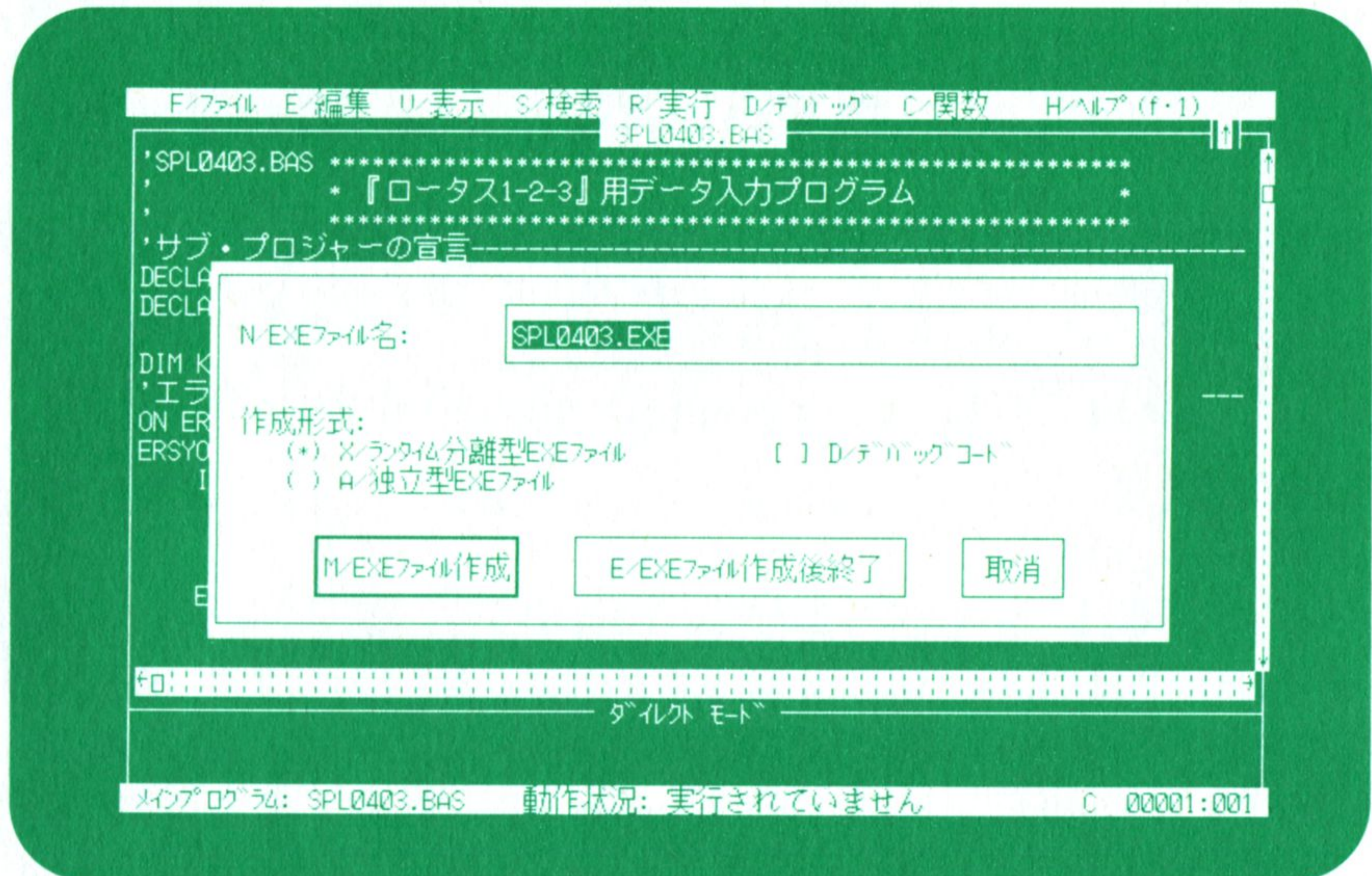
Quick BASICは、統合開発環境といって、メインメニューバーのR/実行を選択して、プルダウンメニューからX/EXEファイル作成の項目を選ぶだけで、実行可能ファイルの作成ができます。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
'SPL0403.BAS ******************  S/スタート          SHIFT+F5  ***
'            * 『ロータス1-2-3』  R/リスタート                  *
'            ******************  N/継続                   F5  ***
'サブ・プロジャーの宣言-----------  X/実行ファイル作成...
DECLARE SUB APPN (FILNAME$)
DECLARE SUB IPPN (FILNAME$)

DIM KOMOK$(35), INDAT$(35): NON = 0: YAN = 1
'エラー処理 ------------------------------------------------------
ON ERROR GOTO ERSYOL
ERSYOL:
    IF ERR = 53 THEN
            SK = 1
            RESUME NEXT
```

EXEファイル作成を選択すると、ダイアログボックスと呼ばれる画面が表示され、実行用ファイル名の入力待ちになります。

EXEファイルを作るドライブの指定を忘れないように注意してください。

```
 F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  O/オプション        H/ヘルプ
                                   SPL0403.BAS
'SPL0403.BAS ***********************************************************
'            * 『ロータス1-2-3』用データ入力プログラム                    *
'            ***********************************************************
'サブ・プ ┌──────────── X/実行ファイル作成 ────────────┐
DECLARE SU │ 実行ファイル名とコンパイルスイッチを設定してください │
DECLARE SU │                                                     │
DIM KOMOK$ │ N/実行ファイル名:  [C:¥ITI05¥SPL0403.EXE        ]    │
'エラー処  │                                                     │
ON ERROR G │ [X] D/デバッグ                                       │
ERSYOL:    │                                                     │
    IF ERR │ < M/実行ファイル作成 >  < E/実行ファイル作成後終了 >  │
       ON  ├─────────────────────────────────────────────────────┤
   ELSEIF  │                          < 取消 >  < H/ヘルプ >      │
           └─────────────────────────────────────────────────────┘
──────────────────────── ダイレクト モード ────────────────────────

F1=ヘルプ  RETURN=実行  ESC=取消  TAB=次項目  カーソルキー=選択  :  C  00001:001
```

Quick BASIC V4.2では、ランタイム分離型EXEファイルか、分離型EXEファイルかを選択できます。ファイル名を入力後、[TAB]キーでカーソルをジャンプさせるか[GRPH]+[X]、あるいは[A]を使います。

[リターン]キーを押すと、コンパイラが起動します。

```
F/ファイル E/編集 V/表示 S/検索 R/実行 D/デバッグ C/関数 H/ヘルプ(f･1)
SPL0403.BAS
'SPL0403.BAS ***********************************************
'            * 『ロータス1-2-3』用データ入力プログラム      *
'            ***********************************************
'サブ・プロジャーの宣言--------------------------------------
DECLA
DECLA   N/EXEファイル名:  SPL0403.EXE

DIM K
'エラ   作成形式:
ON ER     (*) X/ランタイム分離型EXEファイル     [ ] D/デバッグコード
ERSYO     ( ) A/独立型EXEファイル
    I

    E     N/EXEファイル作成   E/EXEファイル作成後終了   取消

ダイレクト モード
メインプログラム: SPL0403.BAS   動作状況: 実行されていません   C 00001:001
```

コンパイルが終了したら、Quick BASICがなくてもプログラムの実行が可能となります。

```
A:¥>ITI05¥SPL0403
```

```
                                        拡張子は自動的に.PRNです
項目数をいれてください（30項目まで:不明の時は、99）＝　99
最後に　//　を入力してください
 1 番目の項目は＝　？データ1
 2 番目の項目は＝　？データ2
 3 番目の項目は＝　？データ3
 4 番目の項目は＝　？データ4
 5 番目の項目は＝　？//

『◇ データ入力を行います（END=E// BACK=B//）◇』
検体番号＝　 1 番
 1）データ1＝ えんどうけんいち
 2）データ2＝ 遠藤謙一
 3）データ3＝ 東京都渋谷区代々木
 4）データ4＝ TEL03-123-4567

検体番号＝　 2 番
 1）データ1＝ なかむらはるこ
```

5．4　ロータス1-2-3でデータを使う

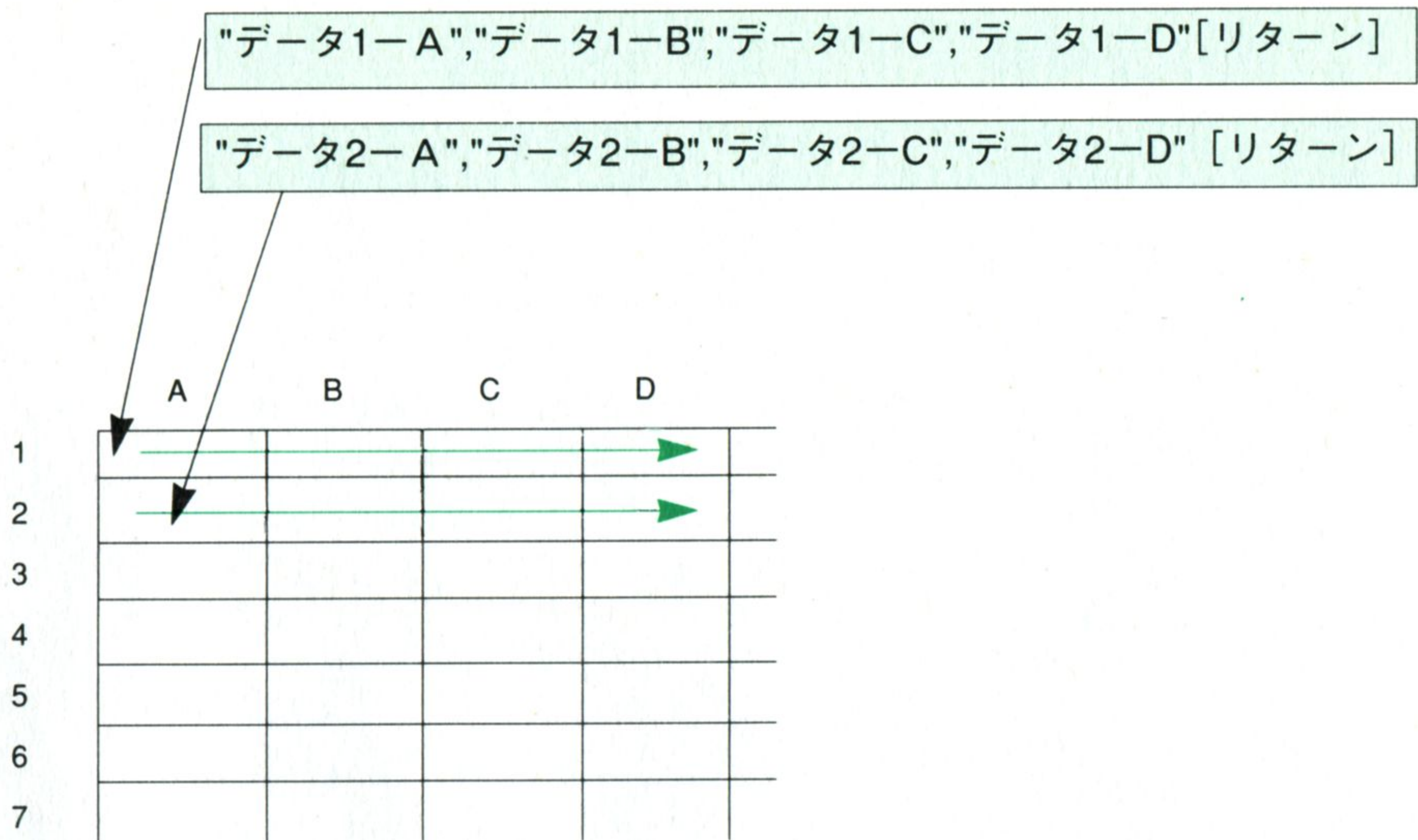
Quick BASICのシーケンシャルファイルをロータス1-2-3で使える形にして保存します。拡張子は自動的にDATがつくようにしていますが、PRNのほうがよいかもしれません。

1　ロータス1-2-3で使えるデータの形

Quick BASICに保存されたデータの形は、"データ１","データ２","データ・・"になっています。この形であれば、ロータス1-2-3のカーソルのある位置から、ロータスがテキストファイルデータとして読み込んでくれます。

"データ1ーA","データ1ーB","データ1ーC","データ1ーD"[リターン]

"データ2ーA","データ2ーB","データ2ーC","データ2ーD" [リターン]

	A	B	C	D
1				
2				
3				
4				
5				
6				
7				

データ入力用プログラムで作ったデータ

『ロータス1-2-3』で読み込めるテキストファイル型データとは、

次のように保存されています。

```
B:¥>TYPE RENSYU.PRN
" 4"
"データ1","データ2","データ3","データ4"
"えんどうけんいち","遠藤謙一","東京都渋谷区代々木","TEL03-123-4567"
"なかむらはるこ","中村晴子","福島県左内市2323","TEL0123-45-6789"
"いのうえかずま","井上一馬","佐賀県葉隠郡桑畑蚕2匹","TEL987-6-5432"
"ささきたかお","佐々木孝雄","京都府笑無重巣土須3.3A",""
"かとうなみこ","加藤波子","青森県殿市糸山気RA21","FAX0123-45-6789"

B:¥>
```

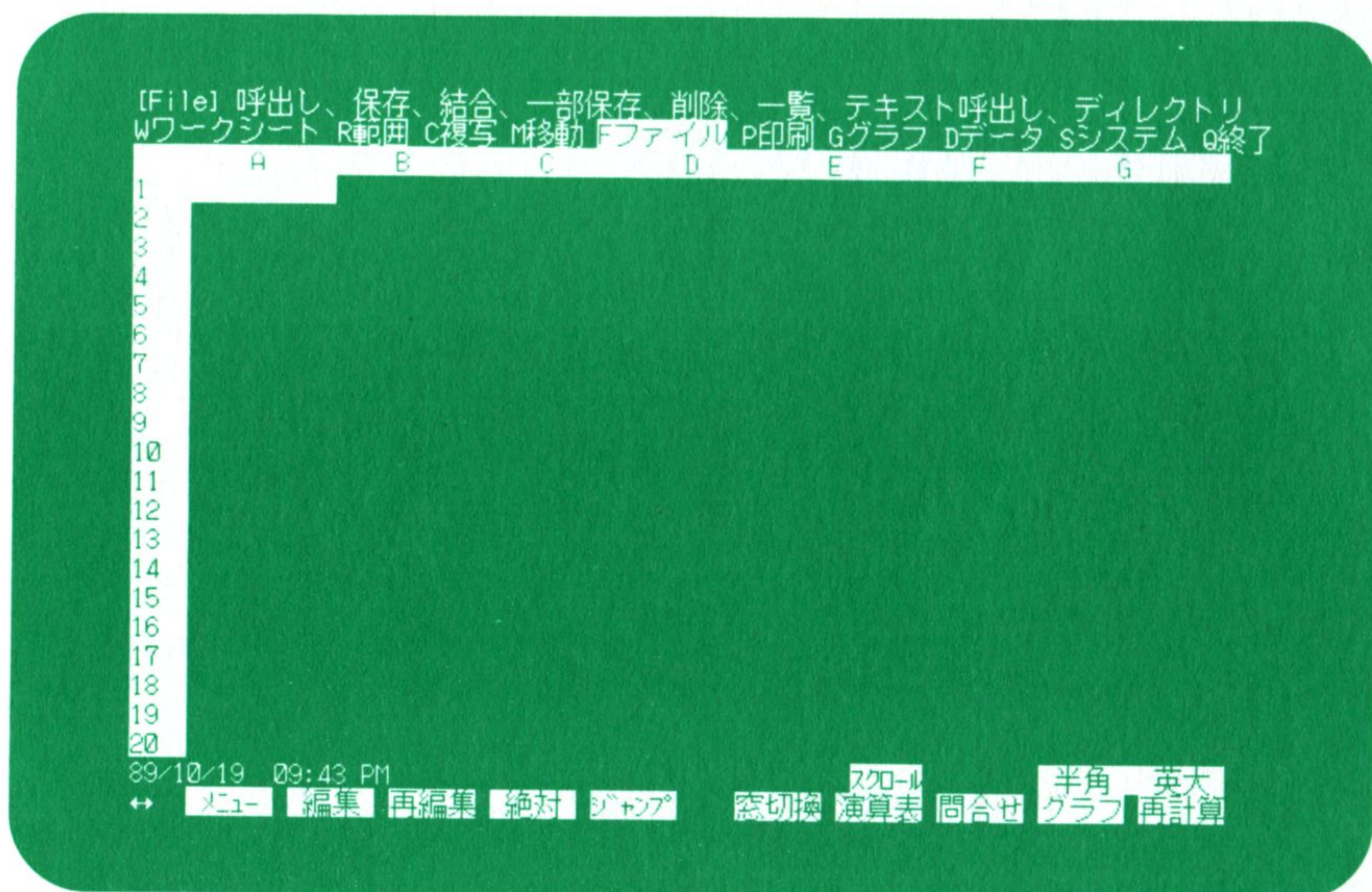
この形式であれば、

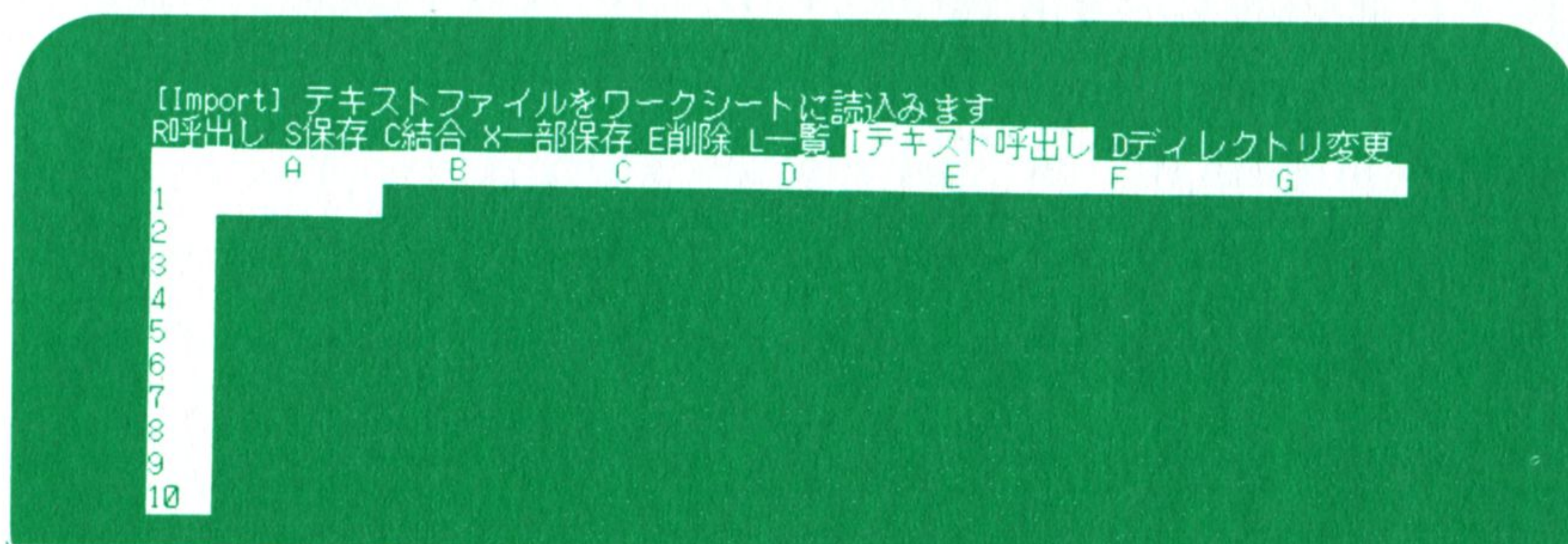
『新松』のテキストファイル型で保存したものや『一太郎』で作ったものは、

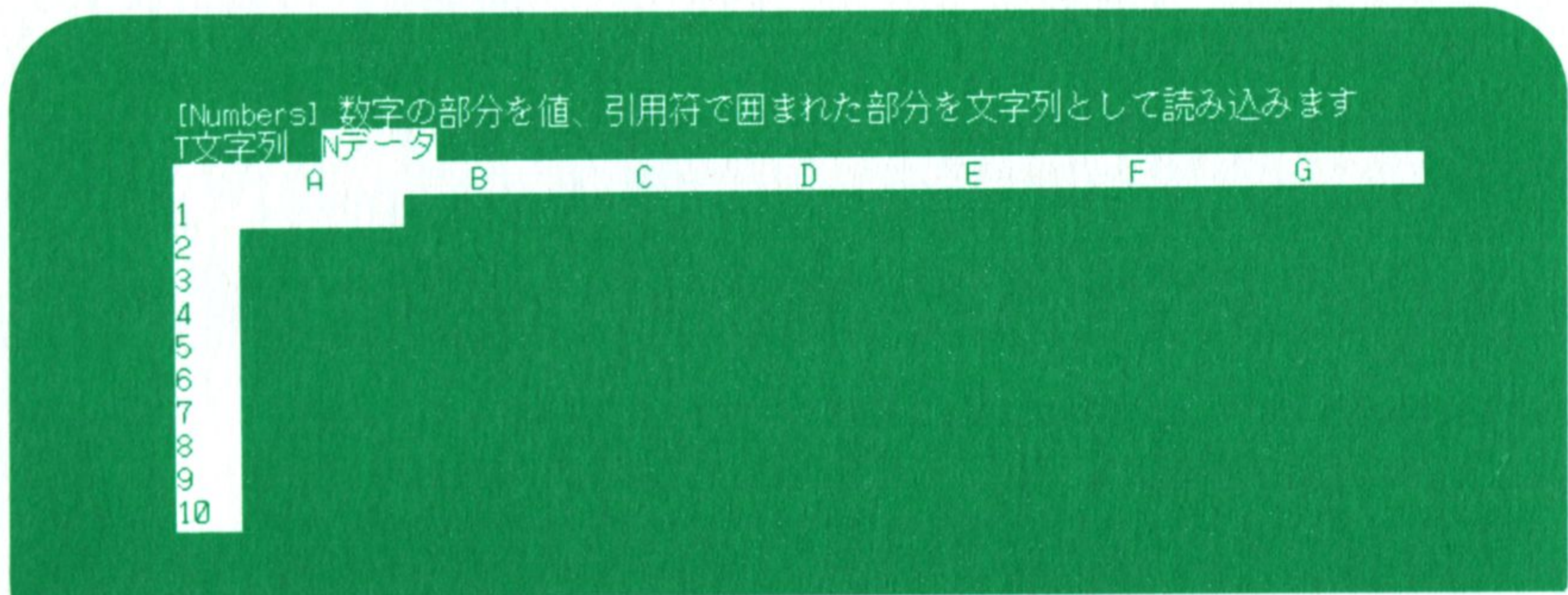
『ロータス1-2-3』で利用できます。

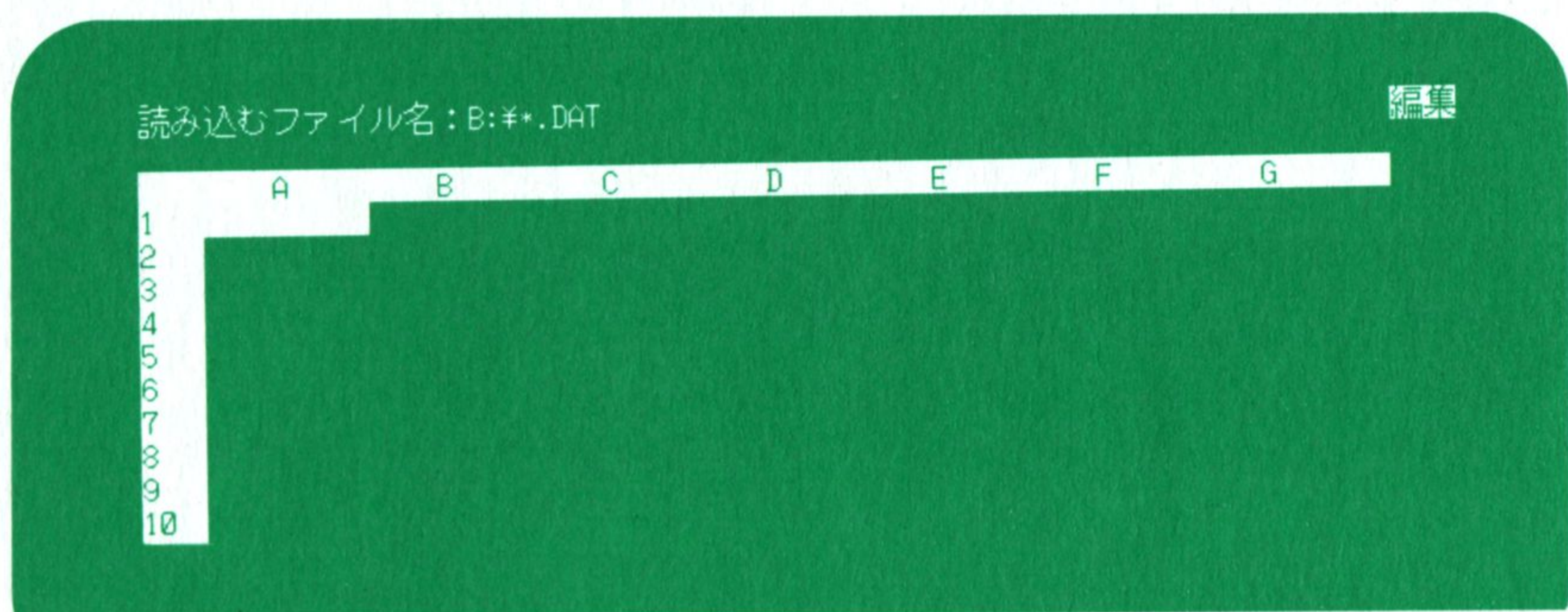
2　ロータスへのデータ読み込み

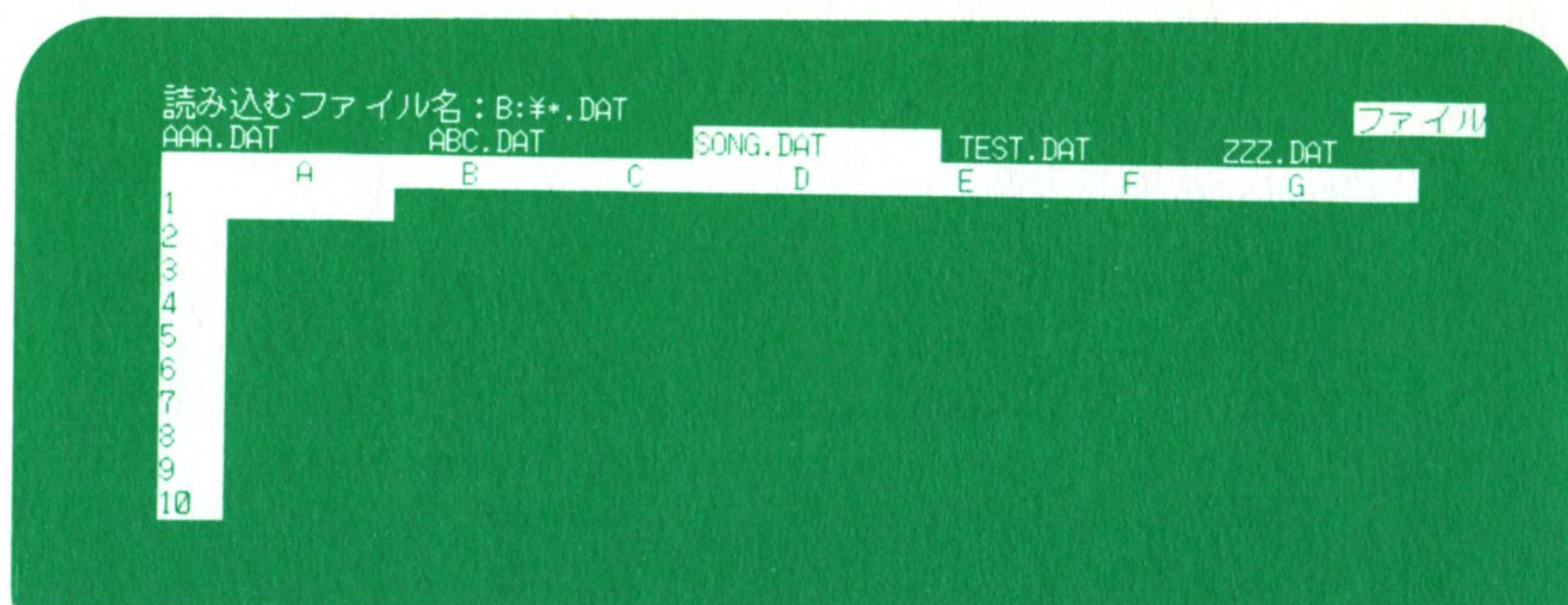
ロータスを起動するとワークシート画面が表示されます。[f･1]を押して、カーソルを画面上部にあるメニューに移動し、カーソルを[←][→]キーで、「Fファイル」に移動させます。

```
[File] 呼出し、保存、結合、一部保存、削除、一覧、テキスト呼出し、ディレクトリ
Wワークシート R範囲 C複写 M移動 Fファイル P印刷 Gグラフ Dデータ Sシステム Q終了
      A         B         C         D         E         F         G
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
89/10/19  09:43 PM
                                                  スクロール           半角   英大
↔  メニュー  編集  再編集  絶対  ジャンプ   窓切換  演算表  問合せ  グラフ  再計算
```

[リターン]キーを押すと、メニューが変わります。カーソルを[←][→]キーで、「Iテキスト呼出し」に移動させます。

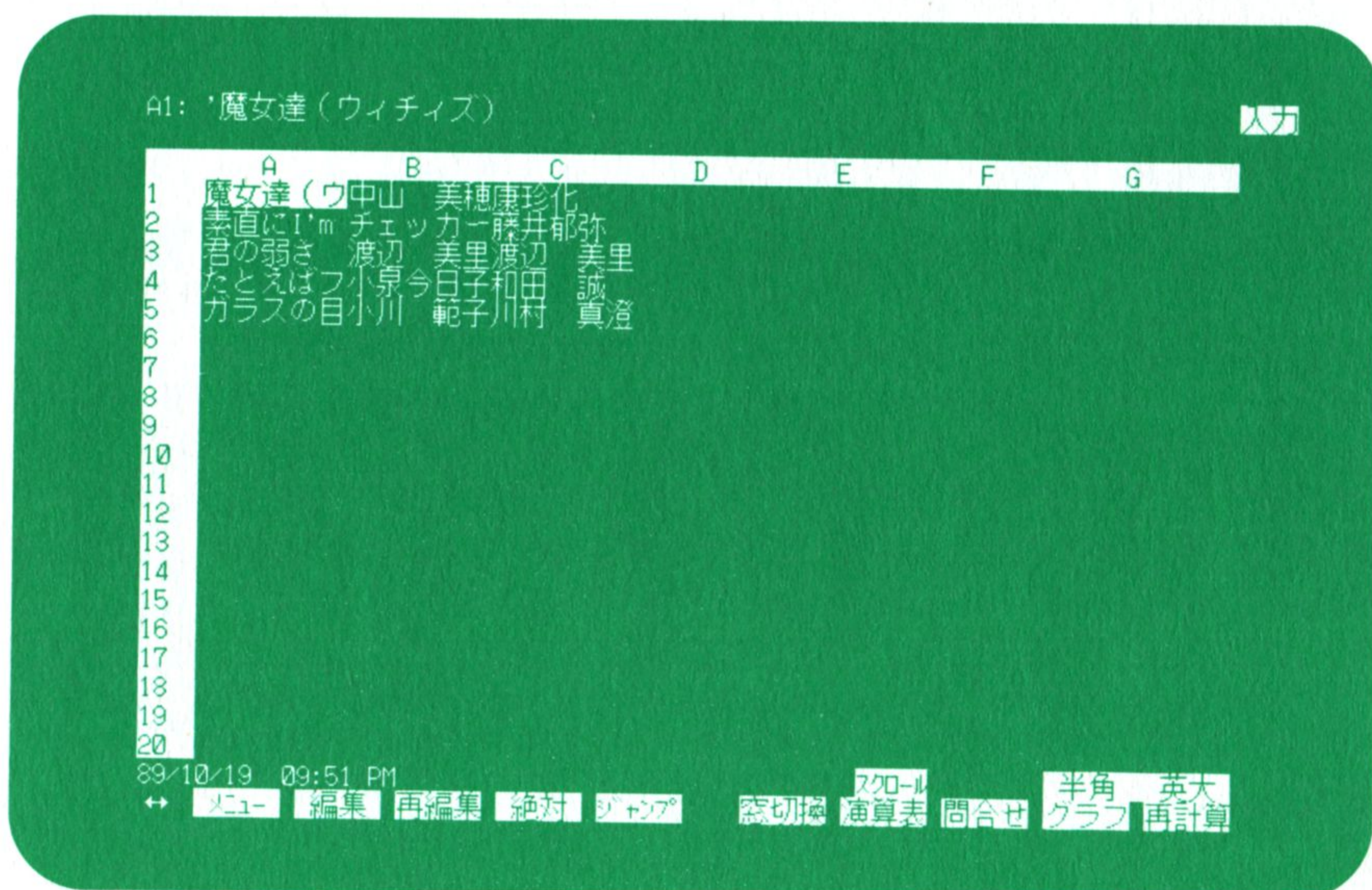
```
[Import] テキストファイルをワークシートに読込みます
R呼出し S保存 C結合 X一部保存 E削除 L一覧 Iテキスト呼出し Dディレクトリ変更
      A         B         C         D         E         F         G
1
2
3
4
5
6
7
8
9
10
```

続いて、[リターン]キーを押すとメニューが変わります。入力しているデータが文字列であっても「Nデータ」のほうを選択します。「T文字列」は、データをセルごとに収めてくれません。

```
[Numbers] 数字の部分を値、引用符で囲まれた部分を文字列として読み込みます
T文字列  Nデータ
         A         B         C         D         E         F         G
1
2
3
4
5
6
7
8
9
10
```

読み込むファイル名を入力します。ワイルドカードが使えますので、*.DATと入力します。このとき、データの入っているディスクの指定を忘れないようにしてください。

```
読み込むファイル名：B:¥*.DAT                                             編集
         A         B         C         D         E         F         G
1
2
3
4
5
6
7
8
9
10
```

拡張子「.DAT」の登録ファイルの一覧が表示されます。カーソルを[←][→]キーで移動して、目的のファイル名を選択します。一太郎で、"データ","データ"と保存したテキストファイルであれば、*.JXWとすれば、一太郎のファイル一覧が表示されます。

```
読み込むファイル名：B:¥*.DAT                                    ファイル
AAA.DAT         ABC.DAT         SONG.DAT        TEST.DAT        ZZZ.DAT
        A         B         C         D         E         F         G
1
2
3
4
5
6
7
8
9
10
```

［リターン］キーを押すと、Quick BASICで保存した、「SONG.DAT」のデータがロータスのセルの位置から読み込まれます。

この画面では、セルが20（半角）文字に設定されているので、データの一部しか見ることができません。

```
A1: '魔女達（ウィチィズ）                                          入力
        A         B         C         D         E         F         G
1  魔女達（ウ中山　美穂康珍化
2  素直にI'm チェッカー藤井郁弥
3  君の弱さ　渡辺　美里渡辺　美里
4  たとえばフ小泉今日子和田　誠
5  ガラスの目小川　範子川村　真澄
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
89/10/19  09:51 PM                                スクロール         半角  英大
↔  メニュー  編集  再編集  絶対  ジャンプ   窓切換  演算表  問合せ  グラフ  再計算
```

ロータスのセル幅を24に変更すると、登録したデータの全てを見ることができます。ここでは、セル幅を24にしています。

```
A1: [ﾊﾊﾞ24] '魔女達（ウィチィズ）                                入力
         A                          B                          C
1  魔女達（ウィチィズ）      中山　美穂                  康珍化
2  素直にI'm Sory             チェッカーズ                藤井郁弥
3  君の弱さ                   渡辺　美里                  渡辺　美里
4  たとえばフォーエバー       小泉今日子&真田弘之         和田　誠
5  ガラスの目隠し             小川　範子                  川村　真澄
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
89/10/19  09:55 PM                              ｽｸﾛｰﾙ            半角　英大
↔  ﾒﾆｭｰ  編集  再編集  絶対  ｼﾞｬﾝﾌﾟ    窓切換  演算表  問合せ  グラフ  再計算
```

これ以後は、ロータス1-2-3を使ってデータを加工することになります。

このように、Quick BASICで作ったプログラムを使って入力したデータを「ロータス1-2-3」で使う、あるいは、一太郎で入力した文章を別の表計算ソフトで活用するというのが、データの互換性です。MS-DOSで動くソフトで入力されたデータは、機種が違っても互換性をもっているものが多くあります。一度入力したデータをいろいろなところで、再入力せずに活用できるのもパソコンの魅力です。

おまけ　ショートプログラム　2進数変換プログラム

```
'SPL4-04.BAS
CLS
 DIM B(17)
 DO
        N = 2
    INPUT "16進数&Hxxxx 10進数そのまま 終了(END)=", IN$
           IF IN$ = "END" OR IN$ = "end" OR IN$ = "ｲﾐｼ" THEN END
           I = VAL(IN$): A = I
```

```
                    B(1) = I MOD 2
      DO
                B(N) = A ¥ 2
                A = B(N)
                IF A >= 2 THEN B(N) = A MOD 2
                N = N + 1
      LOOP UNTIL A < 2
                    PRINT "10進数＝"; I
                    PRINT "16進数＝&H" + HEX$(I)
                    PRINT " 2進数＝"; : COLOR 6
      FOR J = 16 TO 1 STEP -1
                A$ = STR$(B(J))
                A$ = RIGHT$(A$, 1)
                IF J MOD 4 = 0 THEN A$ = " " + A$
                    PRINT A$;
                B(J) = 0
      NEXT
                PRINT : PRINT : COLOR 7
  LOOP
```

```
16進数&Hxxxx 10進数そのまま 終了(END)＝&H16
10進数＝ 22
16進数＝&H16
 2進数＝ 0000 0000 0001 0110

16進数&Hxxxx 10進数そのまま 終了(END)＝100
10進数＝ 100
16進数＝&H64
 2進数＝ 0000 0000 0110 0100

16進数&Hxxxx 10進数そのまま 終了(END)＝
```

Quick BASIC

2 応用編

第1章　図形を描くための基本的な命令

1.1　グラフィックスのおまじない

BASICの優れた機能の1つがこのグラフィックスです。PC-9801の画面を構成する縦400、横640の点の集まりを使って、線を引いたり円を描いたりします。

とりあえず、理屈抜きでグラフィックスを楽しむためには、プログラムの最初にSCREEN命令で640×400の画面モードを設定して、画面を消去するCLS文を使います。この2つは、グラフィックを始めるときのおまじないです。

1　グラフィック画面モードのおまじない

PC-9801の画面は、文字を入力するための「テキストモード」と画面上の好きな点をディスプレイ上に光らせる「グラフィックモード」の2つがあります。これらの画面の選択を行なうのがSCREEN命令です。SCREEN命令には次頁の4つがあります。SCREEN 87では、キーボードから文字の入力ができません。

次頁の表から、640×400のディスプレイであれば、SCREEN 0か、SCREEN 88を使えばよいことがわかります。本書では、画面モードのおまじないはSCREEN 0を使っていきます。

モード	説　明
SCREEN 81	文字だけを表示する画面モードの指定
SCREEN 84	640×200のグラフィックモードとテキストモードの混在
SCREEN 87	640×400のグラフィック画面だけを指定
SCREEN 0 SCREEN 88	640×400のグラフィックモードとテキストモードの混在

2　画面消去のおまじない

画面消去はCLSを使います。CLSも4種類のモードをもっています。くわしくはリファレンスマニュアルを参照してください。Quick BASICで作ったプログラムに日本語フロントエンドプロセッサを組み込んで使うことを前提にする場合は、CLS 0でなければなりません。CLSを使うと、画面の25行目で日本語フロントエンドプロセッサのシステムラインとQuick BASICのファンクション表示がぶつかり合って画面が乱れることがあります。

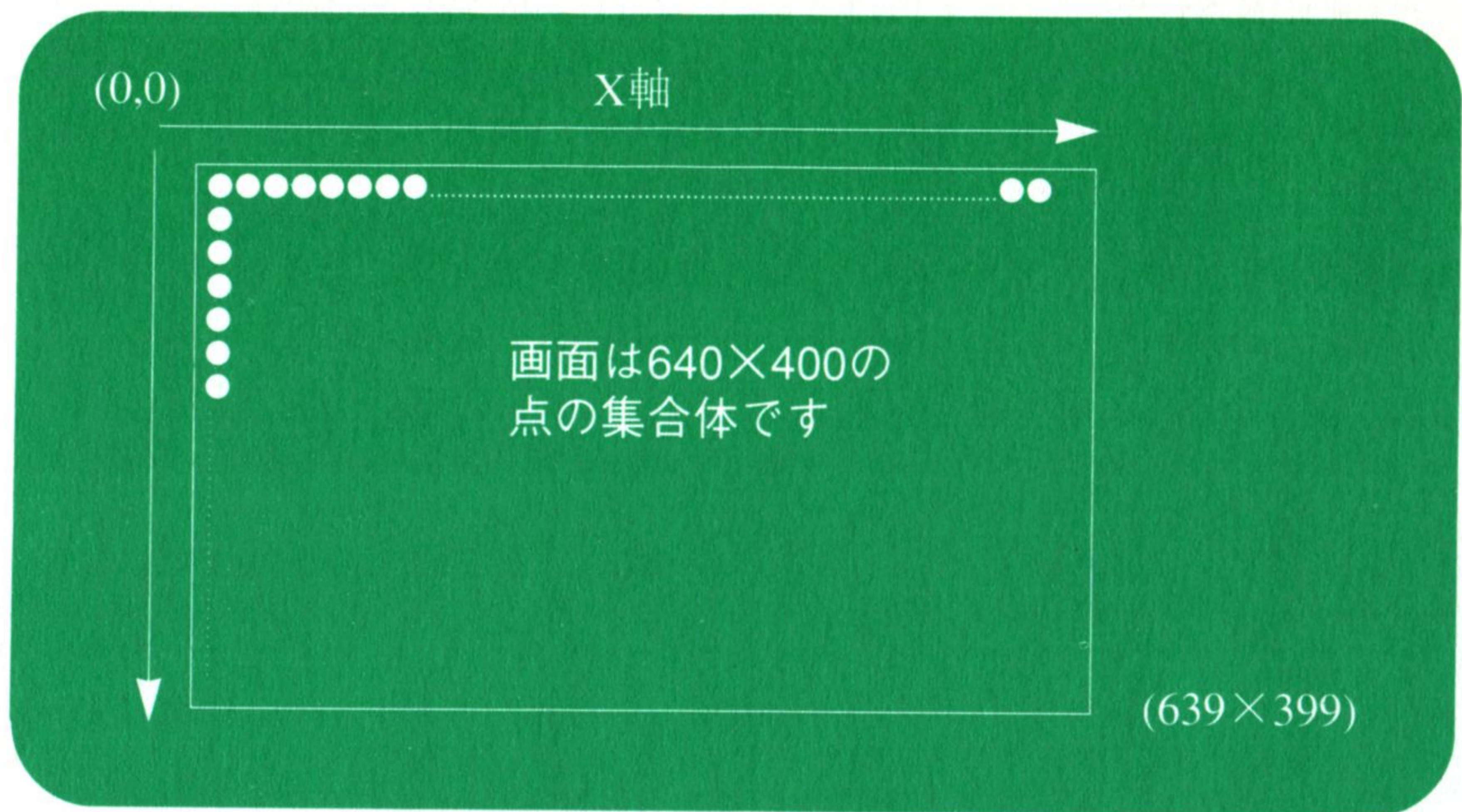
1.2　画面に点を表示する

PC-9801の画面は、縦400と横640の点、つまり、256,000個の点から構成されているわけですが、この256,000個の中心は、どこでしょうか。点の位置は、画面左上のX軸0点、Y軸0点から始まって、画面右下のX軸639、Y軸399となります。

このことから、画面の中心は、X軸が320、Y軸が200です。画面に点を表示する命令は、PSETですから、グラフィックのおまじないを書いたあと、PSETを書き、X軸とY軸の座標点を書いて括弧でくくればおしまいです。

(0,0)　X軸

画面は640×400の
点の集合体です

(639×399)

```
'GR-PR001
CLS
SCREEN 0
 PSET (320, 200)
END
```

プログラムを入力して、[SHIFT]キーを押しながら[f･5]キーを押すと、Quick BASICの編集画面が消えて、「適当なキーを押してください」と表示がされます。あわてて、キーを押さないように！　画面の中央付近をじっと観察してください。これが、画面上に1ドットを表示した状態です。

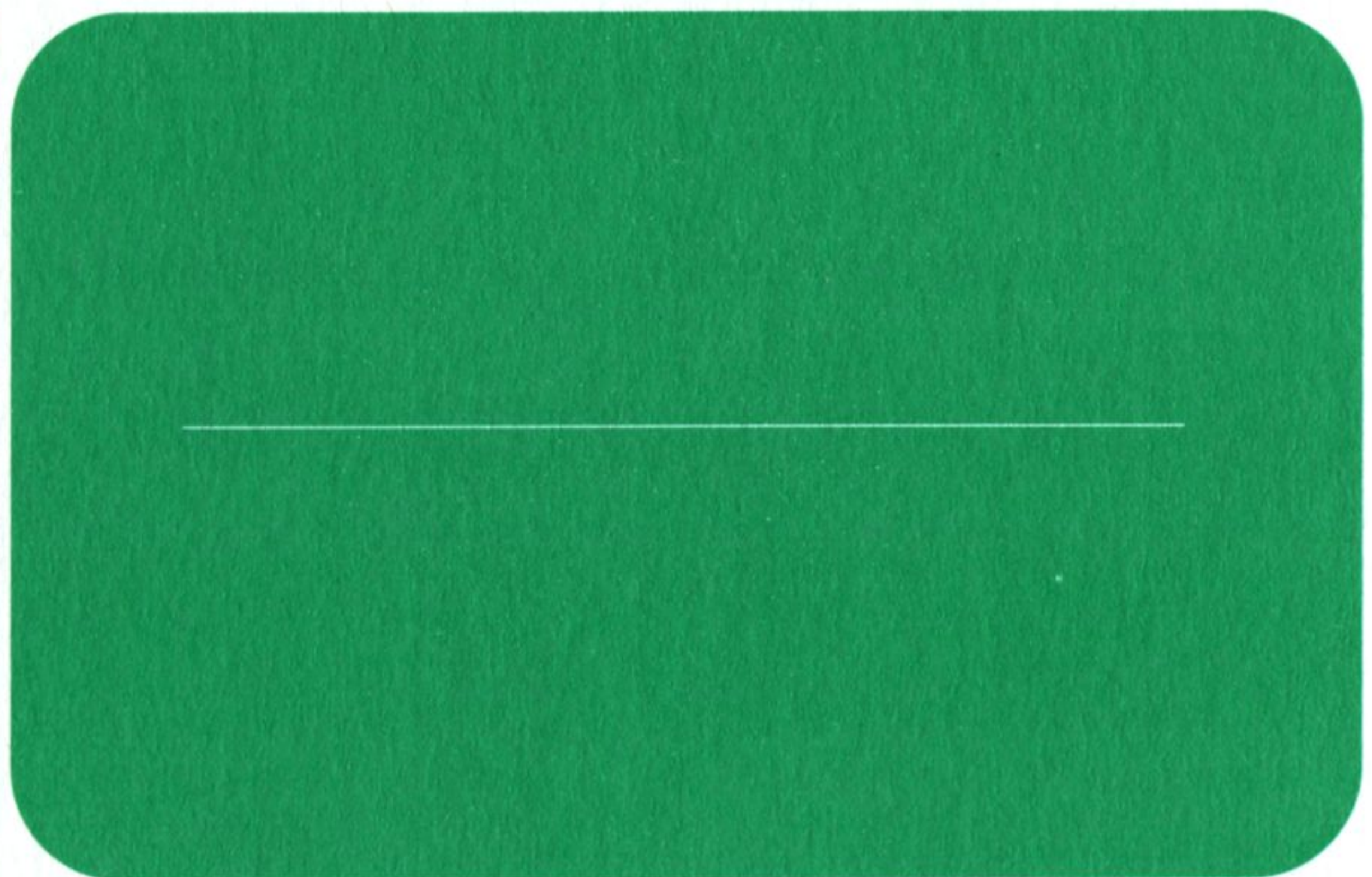
画面に点を表示したのなら、画面の点を消す命令があってよさそうです。素直にそう思ってください。

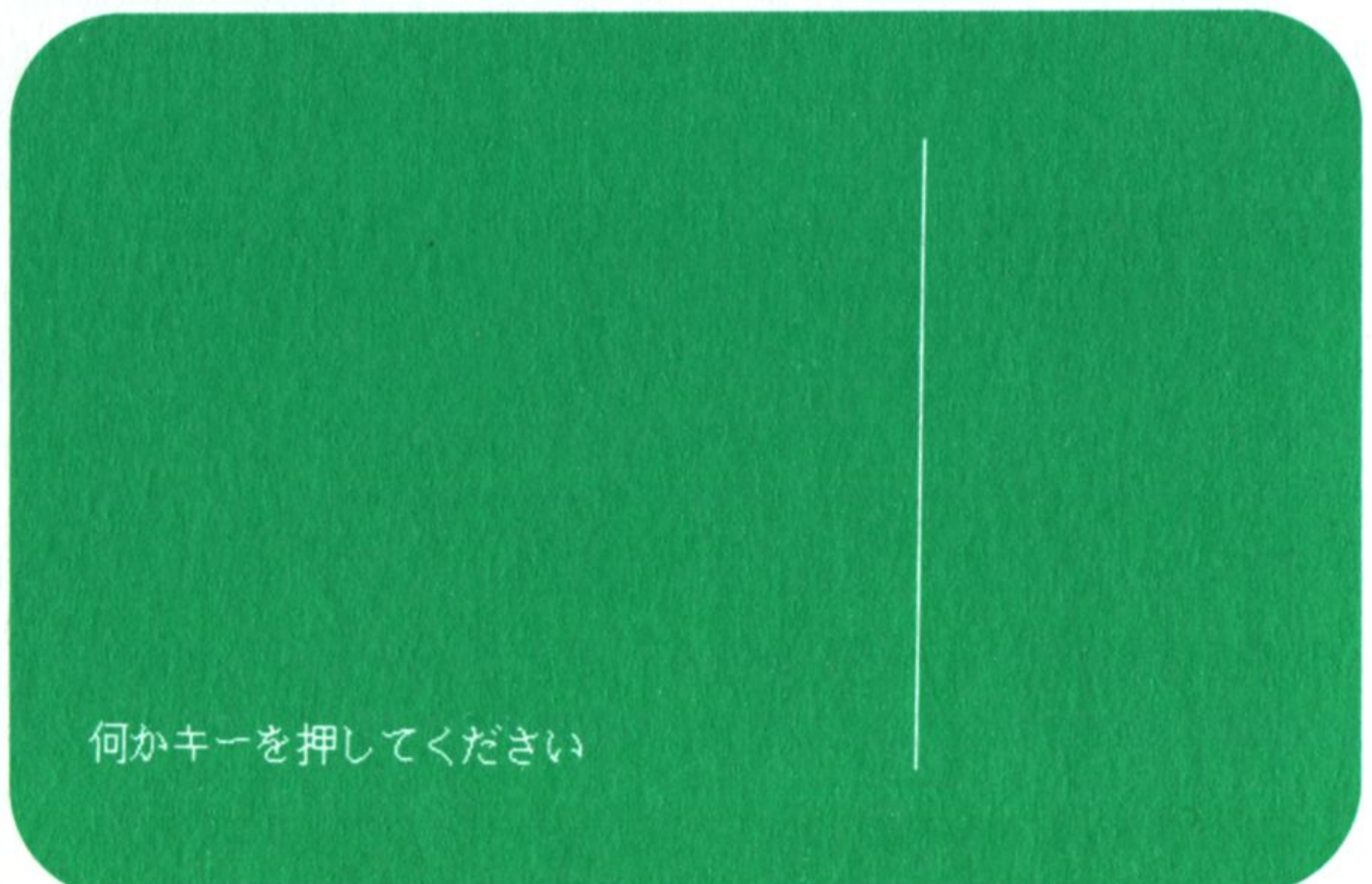
```
'GR-PR002
CLS
SCREEN 0
 PRESET (320, 200)
END
```

１.３　直線の表示

1　点が並ぶと線

点が1列に並んだのが線です。線が4つの方向に曲がって始点に戻ってきたら四角形です。このプログラムは、第1部で学習した繰り返しの命令、FOR･･･NEXTを使って、画面の中央をX軸に添って左から右に点が伸び、やがて点が消えてゆきます。

```
'GR-PR003
 SCREEN 0
  FOR X = 0 TO 639
    PSET (X, 200)
 NEXT
 FOR X = 0 TO 639
    PRESET (X, 200)
 NEXT
END
```

こんどは、Y軸に添って点が伸び、やがて消えてゆくプログラムが次の例です。

```
'GR-PR004
 SCREEN 0
 FOR Y = 0 TO 399
    PSET (320, Y)
 NEXT
 FOR Y = 0 TO 399
    PRESET (320, Y)
 NEXT
END
```

画面中心に点を表示したあと、ホップ・ステップ・ジャンプとばかりに、通信から、X軸に20ドット、Y軸に20ドットプラス側、つまり、点の位置から画面右下方向に、新しい点を表示します。これは、基準となる画面の中央位置から相対的にプラス側にステップしていますから、(20,20)の位置指定のことを相対位置といいます。また、画面中央を示している(320,200)のことを絶対位置と呼んでいます。

```
'GR-PR005
 SCREEN 0
PSET (320, 200)
     PSET STEP(20, 20)
END
```

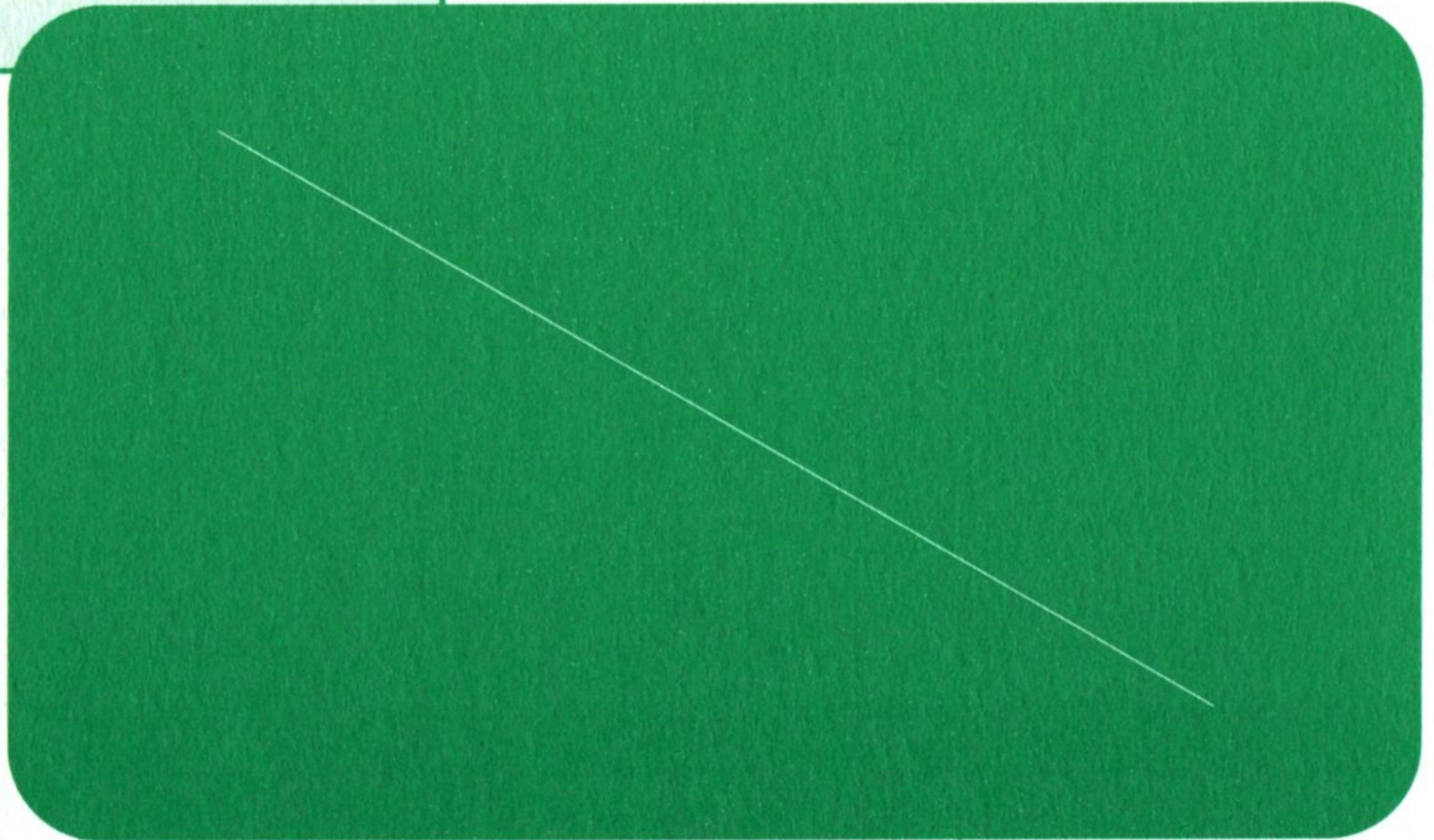
STEP(20,20)

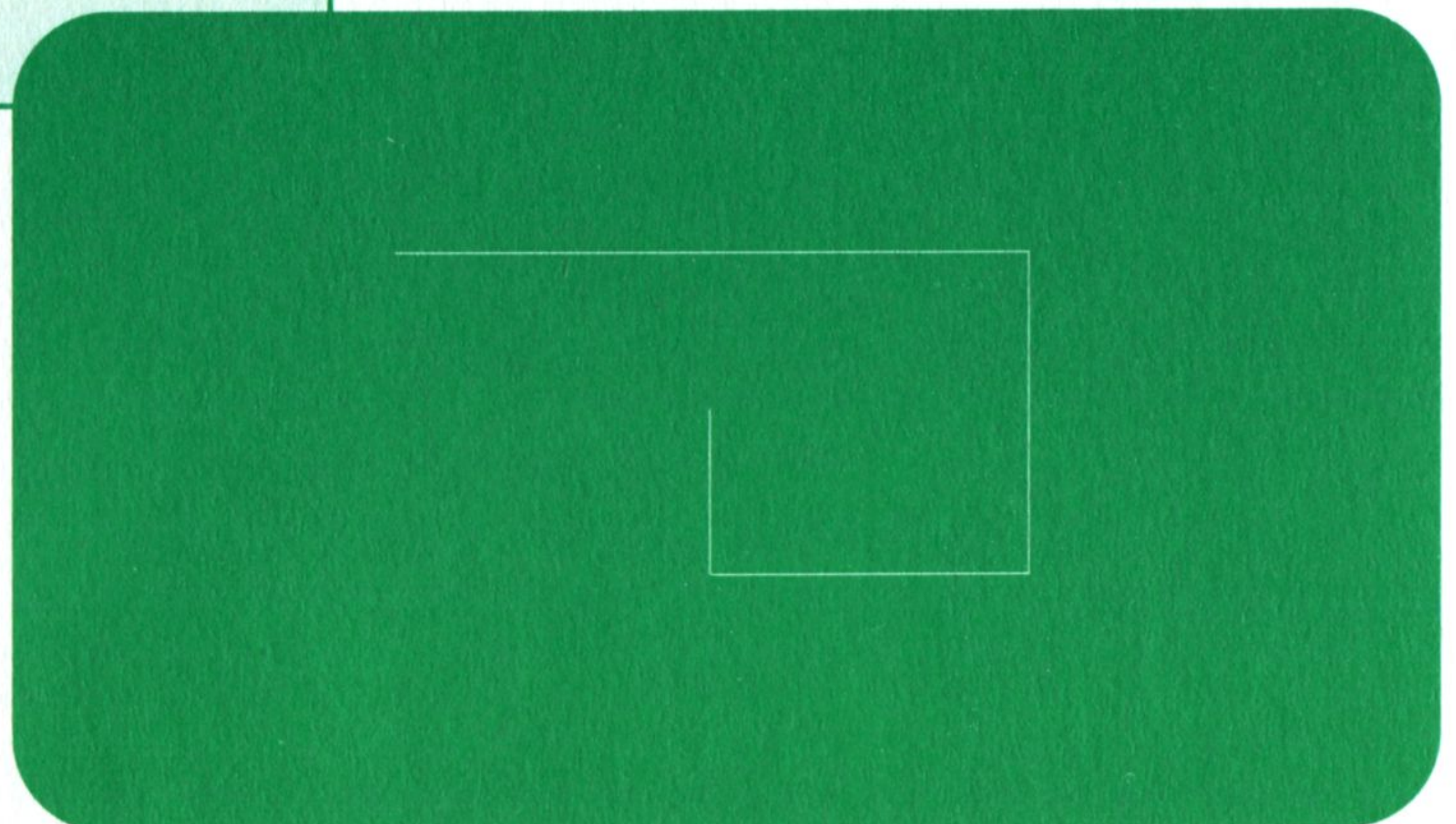
(320,200)

2 直線を引く命令を使う

PSETを使って、点の表示位置を変えることで直線を引きました。ところが、BASICには、直線を引くLINE文という命令をもっています。

この命令は、2点を指定するだけで簡単に直線を引くことができます。画面上、左上から右下に斜めの線を引くプログラムが、次の例です。

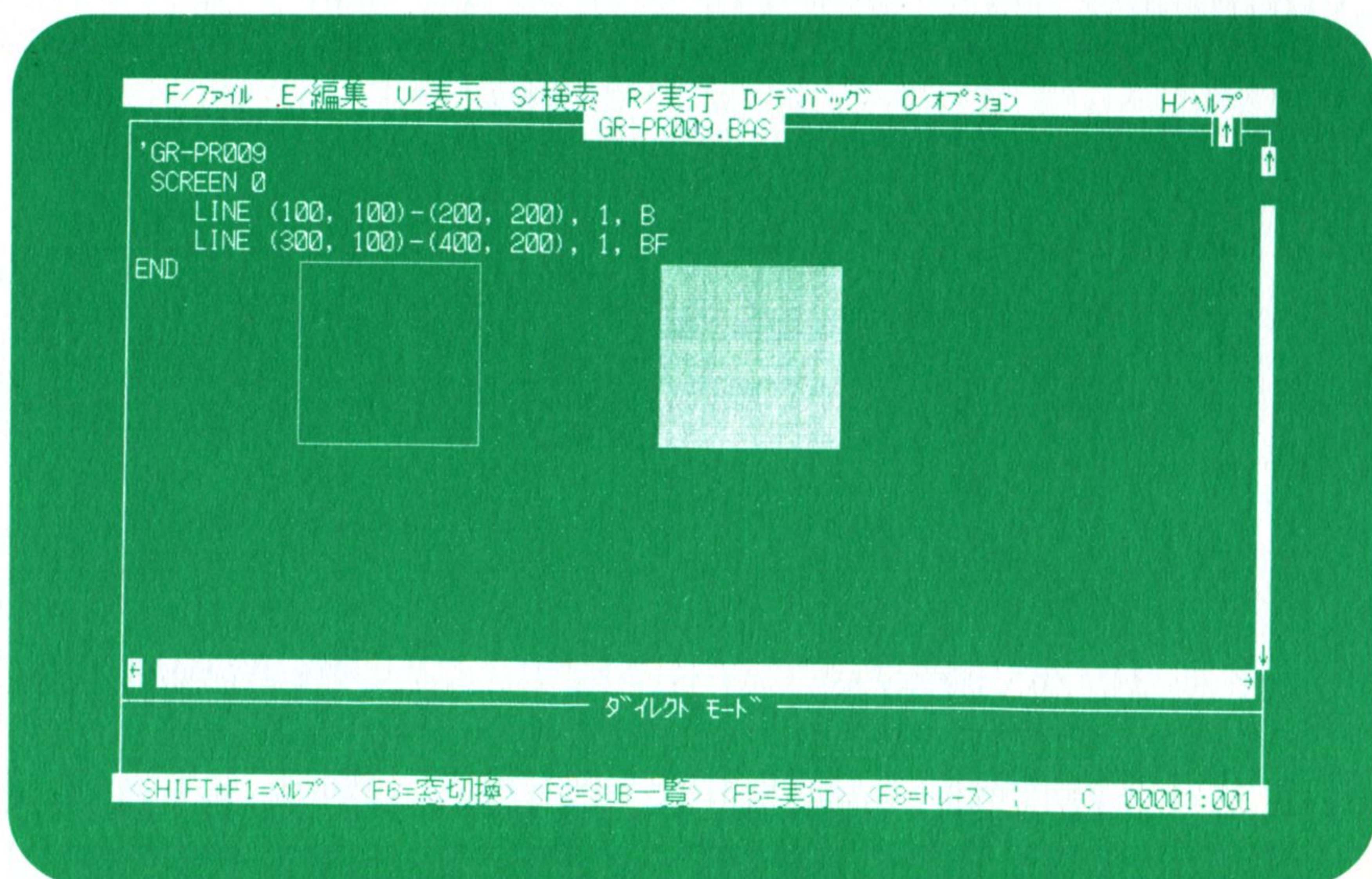
```
'GR-PR006
 SCREEN 0
     LINE (0, 0)-(640, 399)
END
```

こんどは、画面中心に点を表示したあと、LINE文を使って相対位置に線を伸ばしていきます。

```
'GR-PR007
 SCREEN 0
    PSET (320, 200)
     LINE -STEP(0, 50)
     LINE -STEP(100, 0)
     LINE -STEP(0, -100)
     LINE -STEP(-200, 0)
END
```

1.4 四角形の線と面

線を引く命令がLINE文でした。このLINE文にBox（箱）オプションをつけると四角形を描くことができます。これは、PSETの点表示を使っても作れます。終点位置を指定したあと、カンマ（,）が2個続くことに注意してください。最初のカンマの間には、色を指定する数値0〜7が入ります。

```
'GR-PR008
 SCREEN 0
    LINE (100, 100)-(200, 200), , B
END
```

さらに、BoxオプションにFill（いっぱい）オプションを加えると四角形が塗りつぶされて、直線4本の四角形から、面に変わります。このときの面は、白一色。もっとカラフルに楽しみたいものです。カンマの間に色コードの1を入れました。

```
F/ファイル E/編集 V/表示 S/検索 R/実行 D/デバッグ O/オプション H/ヘルプ
GR-PR009.BAS
'GR-PR009
 SCREEN 0
    LINE (100, 100)-(200, 200), 1, B
    LINE (300, 100)-(400, 200), 1, BF
END
ダイレクト モード
<SHIFT+F1=ヘルプ> <F6=窓切換> <F2=SUB一覧> <F5=実行> <F8=トレース> C 00001:001
```

1.5　円を描く

円を描く命令はCIRCLE文です。この命令を使うと真円、楕円、円弧を作ることができます。

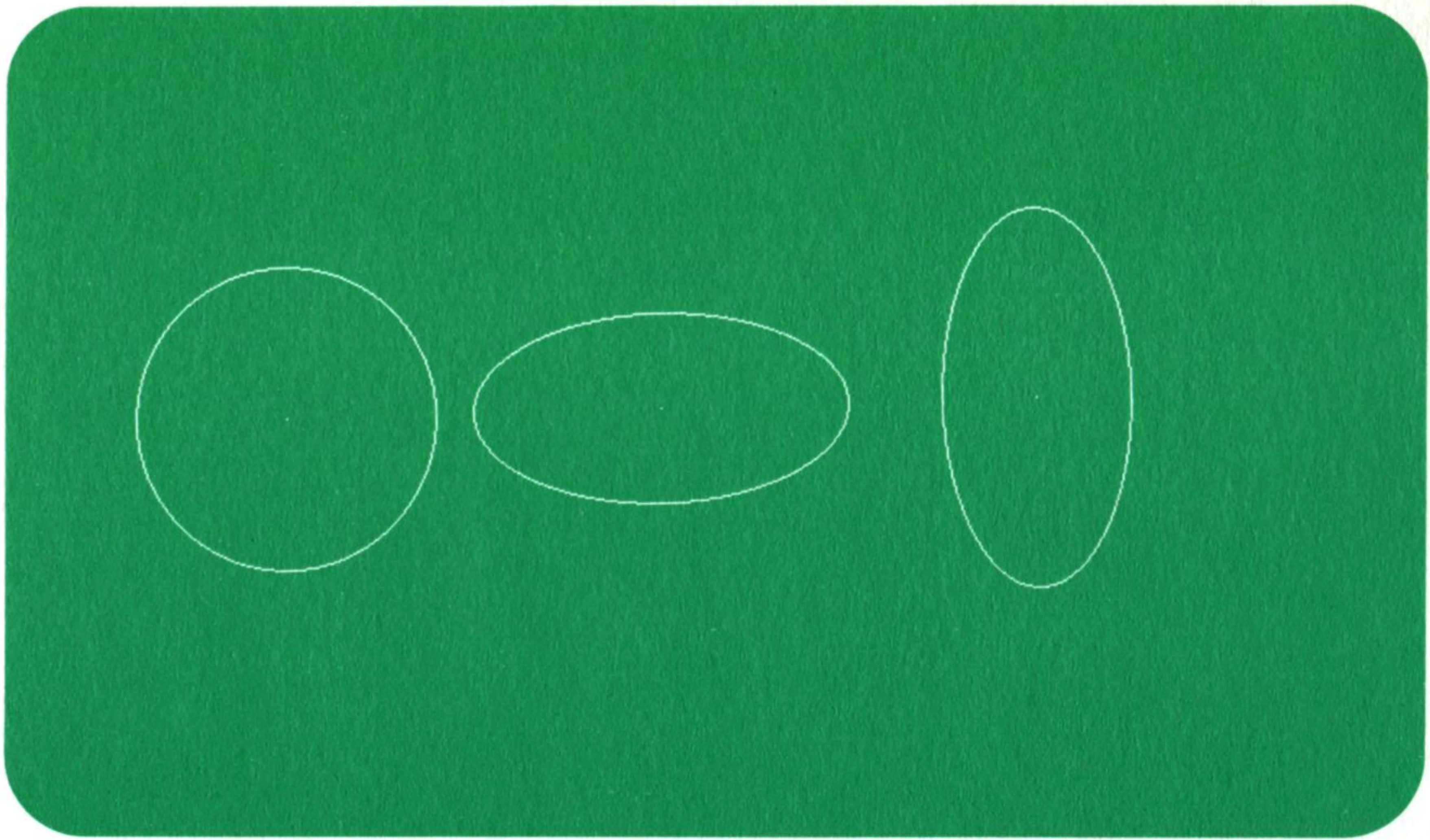
円を描くときは、次のような式になっています。

CIRCLE(X軸の座標,Y軸の座標),半径,色,円のスタート点,終点,円の縦横比

半径は、ドットの数を指定します。

円のスタート、終点は、ラジアンで角度を表わします。詳しくは第3章の曲線を参照してください。

このプログラムは、3つの円が黄色で画面に表示されます。最初の円は真円。つぎは、縦が50、横100の楕円。3つめは、縦が100、横が50の楕円です。

```
'GR-PR010.BAS
 SCREEN 0
 CLS
 X = 100: Y = 200: HANKEI = 80: CL = 6
      PSET (X, Y), CL
      CIRCLE (X, Y), HANKEI, CL

 X = 300: Y = 200: HANKEI = 100: CL = 6
      PSET (X, Y), CL
      CIRCLE (X, Y), HANKEI, CL, , , 1 / 2

 X = 500: Y = 200: HANKEI = 100: CL = 6
      PSET (X, Y), CL
      CIRCLE (X, Y), HANKEI, CL, , , 2 / 1
 END
```

1.6 色をつける

点を表示するPSETや線を描くLINE文にBやFのオプションをつけました。このオプションのことをパラメータといいます。色もカラーコードをもっていて、パラメータとして使用します。N88-BASICのカラーコードとQuick BASICのカラーコードは次のようにちがっていますので注意が必要です。

1 Quick BASICのカラーコード

カラーコード	0	1	2	3	4	5	6	7
N88-BASICの色	黒	青	赤	紫	緑	水	黄	白
Quick BASICの色	黒	青	緑	水	赤	紫	黄	白

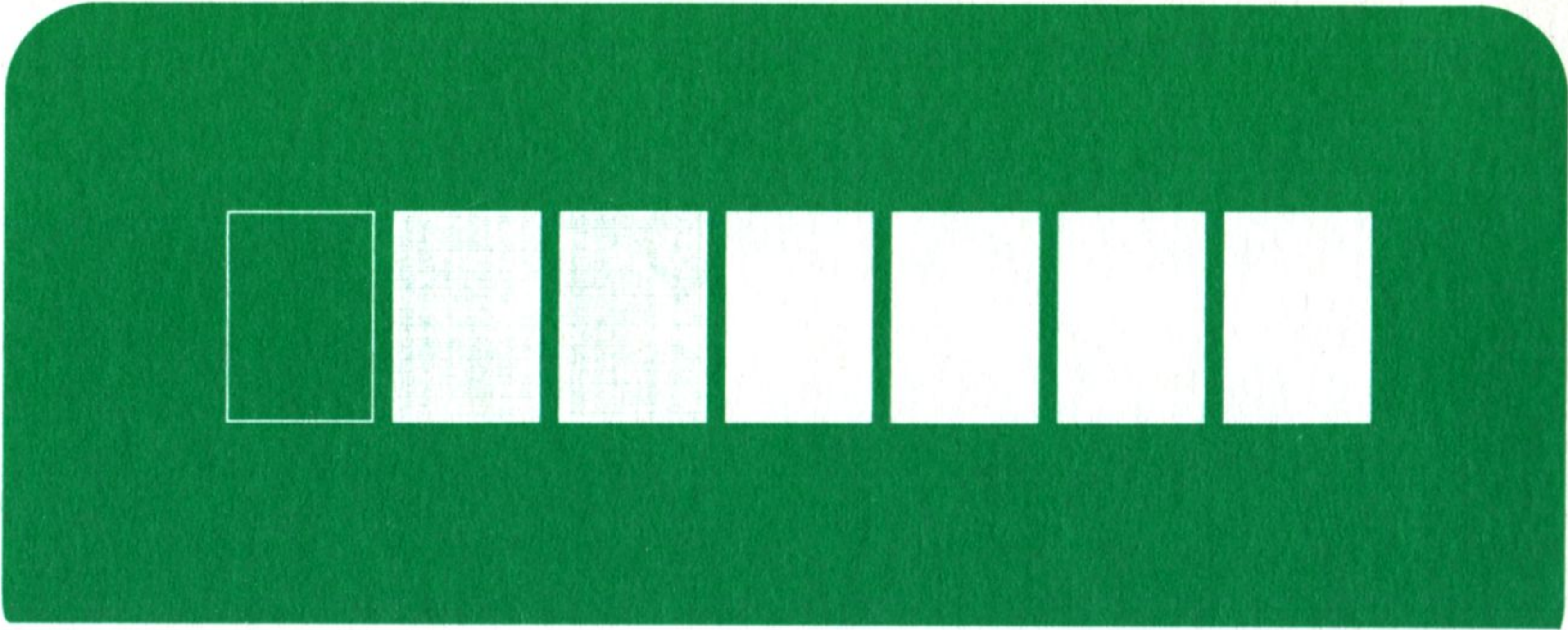
次のプログラムを実行して、色を確認してください。変数CLの中に色コードのパラメータ値が代入されます。

●線の色

```
'GR-PR011.BAS
SCREEN 0
CLS:LOCATE 3, 1
FOR CL = 1 TO 7
    COLOR CL
        PRINT "COLOR･･･" + RIGHT$(STR$(CL), 1)
        PRINT
    LINE (10 * 8, 8 + 2 * (CL * 16))-(78 * 8, 8 + 2 * (CL * 16)), CL
NEXT
```

```
COLOR･･･1 ──────────────────────────
COLOR･･･2 ──────────────────────────
COLOR･･･3 ──────────────────────────
COLOR･･･4 ──────────────────────────
COLOR･･･5 ──────────────────────────
COLOR･･･6 ──────────────────────────
COLOR･･･7 ──────────────────────────
```

●面の色

```
'GR-PR012
SCREEN 0
  CL = 0
   FOR X = 10 TO 490 STEP 80
   LINE (X, 100)-(X + 70, 200), 7, B
   PAINT (X + 1, 101), CL, 7
   CL = CL + 1
NEXT
```

2　色を混ぜる

カラーコードは、光の三原色に基づいています。ＲＧＢというように、レッド（赤）とグリーン（緑）、それにブルー（青）を組み合わせたもので黒を除く6色が決められます。N88-BASICとQuick BASICは、この三色の並び方が違っているため、カラーコードが変わってきます。

N88-BASICの色					Quick BASICの色				
カラーコード	G	R	B	色	カラーコード	R	G	B	色
0	0	0	0	黒	0	0	0	0	黒
1	0	0	1	青	1	0	0	1	青
2	0	1	0	赤	2	0	1	0	緑
3	0	1	1	紫	3	0	1	1	水
4	1	0	0	緑	4	1	0	0	赤
5	1	0	1	水	5	1	0	1	紫
6	1	1	0	黄	6	1	1	0	黄
7	1	1	1	白	7	1	1	1	白

N88-BASICのカラーコードもF-BASICのカラーコードも同じです。この表から従来のカラーコードは、RGBではなくGRBであることがわかります。RGB対応ディスプレイではなく、GRB対応ディスプレイではないかなどと申すまい。普遍的なカラーコードは、どちらがよいかなどという評価も賢い読者の皆様におまかせするとして、カラーコードは、この光の3原色を2進数で表わします。

水色は、緑と青の混ざったもの、紫は赤と青の混ざったものです。それでは、水色に紫の混ざったものは、どう作ればいいでしょう。

8ドット単位で考えると、X軸方向に［水紫水紫水紫水紫］と画面上に表示すれば、よいことがわかります。カラーコード表から水色は011、紫は101です。

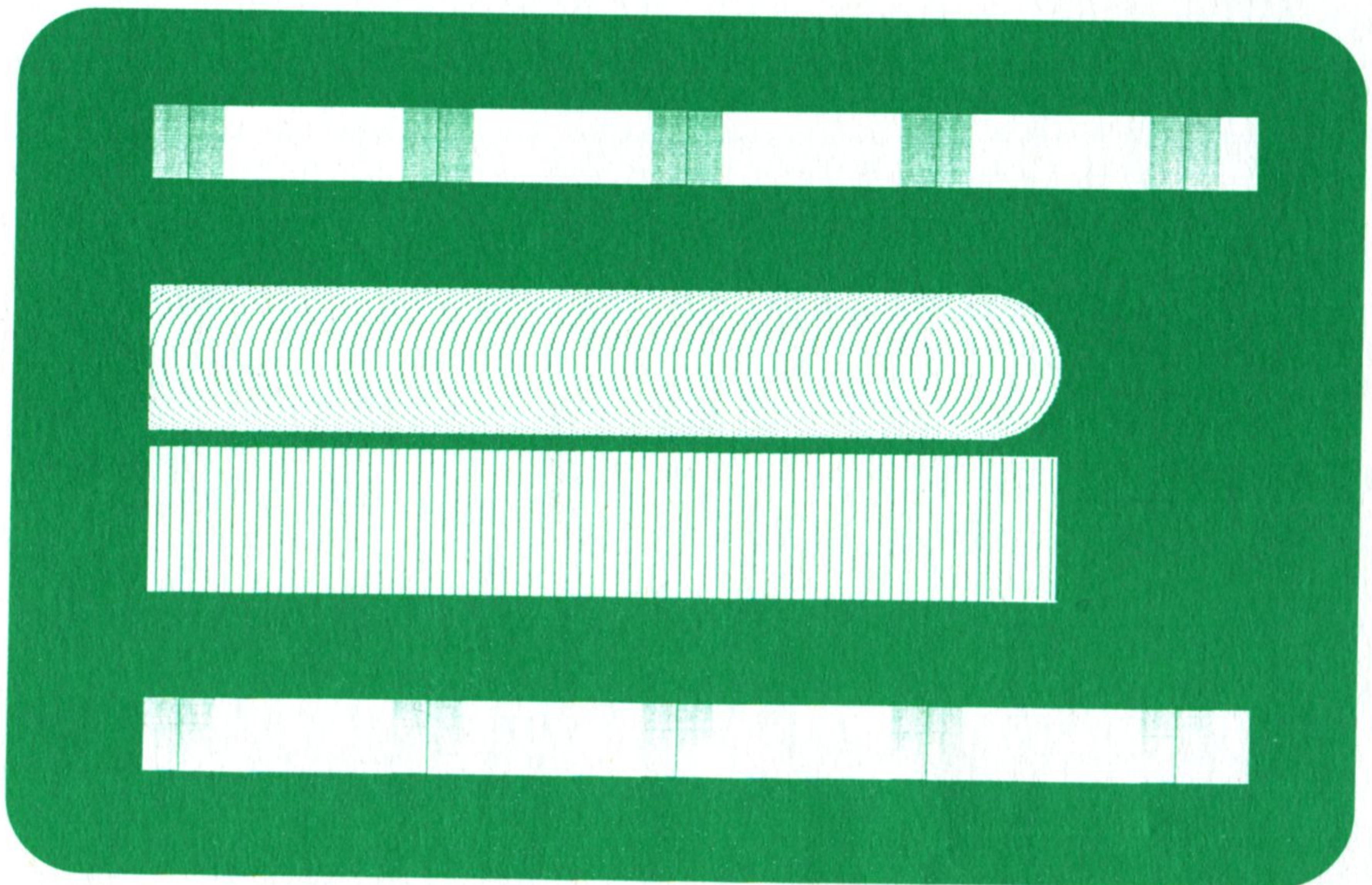
色の混合	水	紫	水	紫	水	紫	水	紫	RGBを16進数
R（赤）	0	1	0	1	0	1	0	1	0101 0101=55H
G（緑）	1	0	1	0	1	0	1	0	1010 1010=AAH
B（青）	1	1	1	1	1	1	1	1	1111 1111=FFH

```
'GR-PR013.BAS
SCREEN 0
CLS
          PAINT (0,0),CHR$(&H55)+CHR$(&HAA)+CHR$(&HFF)
END
```

3　四角形と円の移動

画面上を右から左に固定されたX軸に添って四角形と円を移動させます。色の指定は、変数CLを使うことにします。

```
'GR-PR014
SCREEN 0
CLS
FOR X = 0 TO 600 STEP 20
        CL = CL + 1
        IF CL >= 8 THEN CL = 1
           LINE (X, 10)-(X + 20, 50), CL, BF
           LINE (X, 340)-(X + 20, 380), CL, BF
NEXT
```

```
    FOR CX = 1 TO 600
        CL = CL + 1
        IF CL >= 8 THEN CL = 1
            CIRCLE (CX, 150), 40, CL
            LINE (CX, 200)-(CX + 40, 280), CL, B
    NEXT

FOR CX = 1 TO 600
            CIRCLE (CX, 150), 40, 0
            LINE (CX, 200)-(CX + 40, 280), 0, B
NEXT
            LOCATE 12, 34: PRINT "オシマイ"
END
```

第２章　直線と四角形

2.1　直線のスタイル

直線を引くには、PSETあるいは、LINE文を使います。この2つの命令のバリエーションで線を引くことになります。

LINE文の書式には、線のスタイルを16ビットのパターンで指定することができるようになっています。

```
  LINE(X1,Y1)-(X2,Y2),色,BOX,スタイル
 16ビットパターンは、10進数、16進数で表示します。
■■■■ ■■■■ ■■■■ ■■■■=1111 1111 1111 1111=&HFFFF
□■□■ □■□■ □■□■ □■□■=0101 0101 0101 0101=&HAAAA
□□■■ □□■■ □□■■ □□■■=0011 0011 0011 0011=&H3333
■■□■ ■□■■ □■■□ ■■□■=1101 1011 0110 1101=&HFFFF
□□□■ □□□■ □□□■ □□□■=1111 1111 1111 1111=&HFFFF
'GR-PR201.BAS
 SCREEN 0
  CLS
 LOCATE 2, 5: PRINT "&HFFFF"
   LINE (6 * 16, 2 * 16)-(500, 2 * 16), 7, , &HFFFF
```

```
 LOCATE 4, 5: PRINT "&HEEEE"
   LINE (6 * 16, 4 * 16)-(500, 4 * 16), 1, , &HEEEE
 LOCATE 6, 5: PRINT "&HDDDD"
   LINE (6 * 16, 6 * 16)-(500, 6 * 16), 2, , &HDDDD
 LOCATE 8, 5: PRINT "&HCCCC"
   LINE (6 * 16, 8 * 16)-(500, 8 * 16), 3, , &HCCCC
 LOCATE 10, 5: PRINT "&HBBBB"
   LINE (6 * 16, 10 * 16)-(500, 10 * 16), 4, , &HBBBB
 LOCATE 12, 5: PRINT "&HAAAA"
   LINE (6 * 16, 12 * 16)-(500, 12 * 16), 5, , &HAAAA
 LOCATE 16, 20: PRINT "10進数 100"
   LINE (16, 16 * 16)-(600, 16 * 16), 6, , 100
END
```

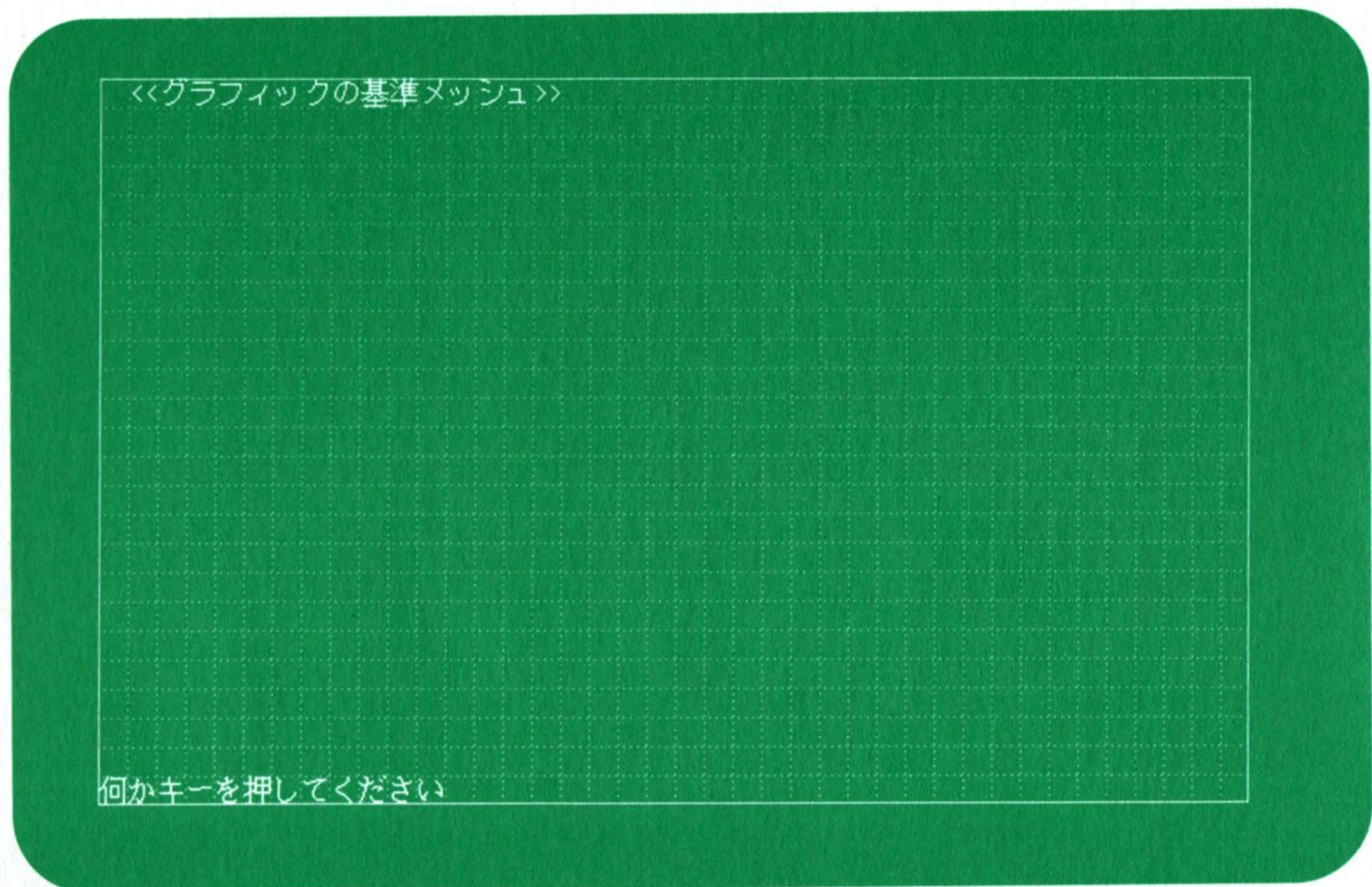
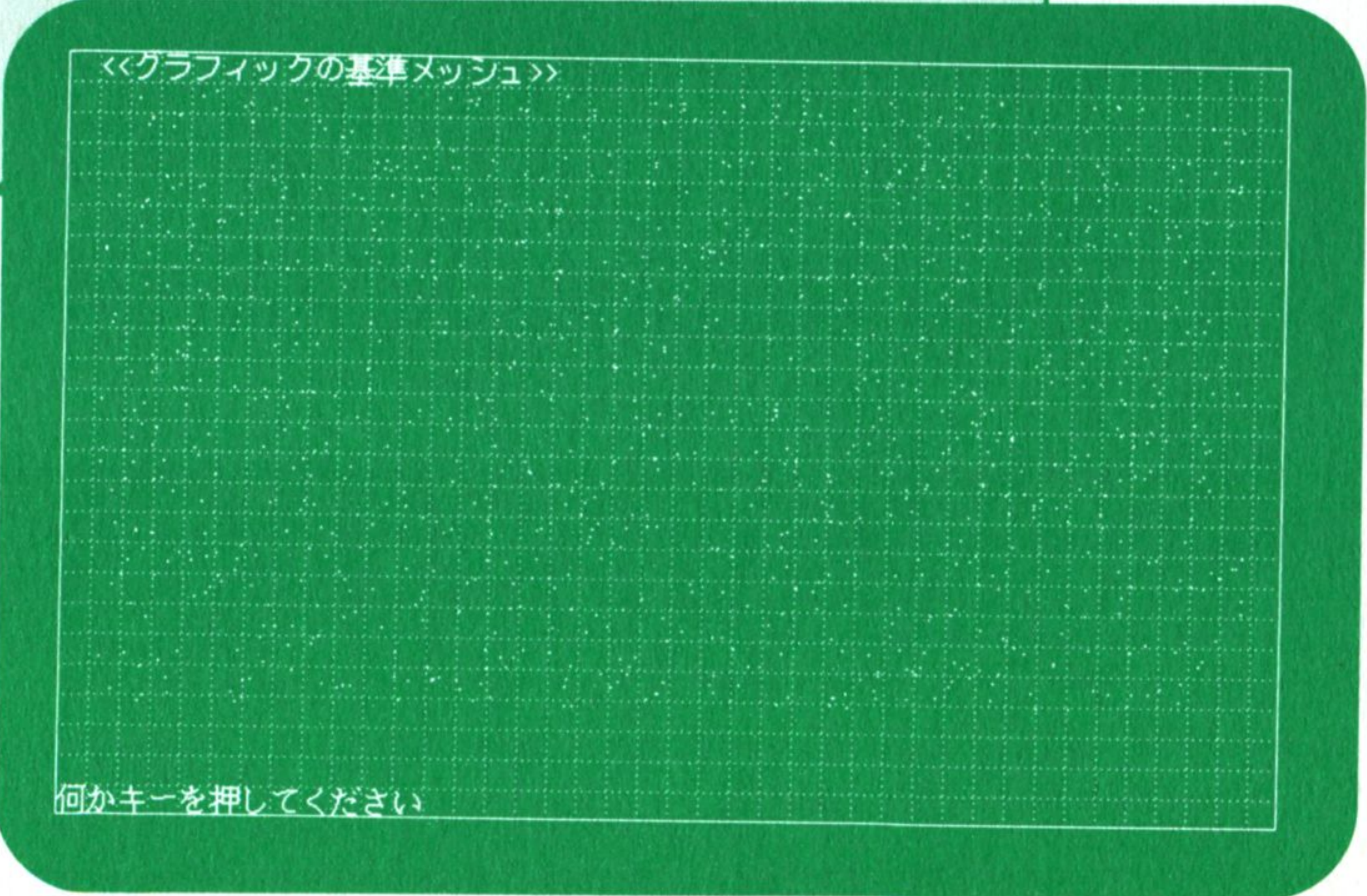
```
&HFFFF
&HEEEE
&HDDDD
&HCCCC
&HBBBB
&HAAAA

10進数 100
```

LINE文の線のスタイルを使って、今後使っていく、グラフィックスのメッシュ画面を作ります。メッシュは、縦、横ともに16ドット間隔の四角をつくることにしました。日本語表示（全角）文字が、ぴったり収まるようになっています。

```
'GR-PR202.BAS
 SCREEN 0
 CLS
   CL = 2
  LINE (0, 0)-(639, 399), 2, B
 FOR X = 0 TO 639 STEP 16
   LINE (X, 0)-(X, 399), CL, , &H1111
```

```
  NEXT
  FOR Y = 0 TO 399 STEP 16
     LINE (0, Y)-(639, Y), CL, , &H1111
  NEXT

  LOCATE 1, 1
         PRINT "  <<グラフィックの基準メッシュ>>"
 END
```

<<グラフィックの基準メッシュ>>

何かキーを押してください

2.2　乱数を使った点と線

メッシュの上で、点と線の表示を乱数で行ないます。乱数を発生させるのは、X軸の値と、Y軸の値です。表示範囲を制限してあります。それが、どこなのかは、自分でプログラムを実行させて確認してください。

1 点の乱数

プログラムは、グラフィックの基準メッシュを省略できます。

```
'GR-PR203.BAS
 RANDOMIZE VAL(RIGHT$(TIME$, 2))
 SCREEN 0
 CLS
    CL = 2
   LINE (0, 0)-(639, 399), 2, B
 FOR X = 0 TO 639 STEP 16
    LINE (X, 0)-(X, 399), CL, , &H1111
 NEXT
 FOR Y = 0 TO 399 STEP 16
    LINE (0, Y)-(639, Y), CL, , &H1111
 NEXT
 LOCATE 1, 1
        PRINT "  <<グラフィックの基準メッシュ>>"
'点の発生
 CL = 7
 FOR I = 0 TO 1000
        X1 = INT(RND(1) * 600) + 20
        Y1 = INT(RND(1) * 320) + 20
    PSET (X1, Y1), CL
 NEXT
 END
```

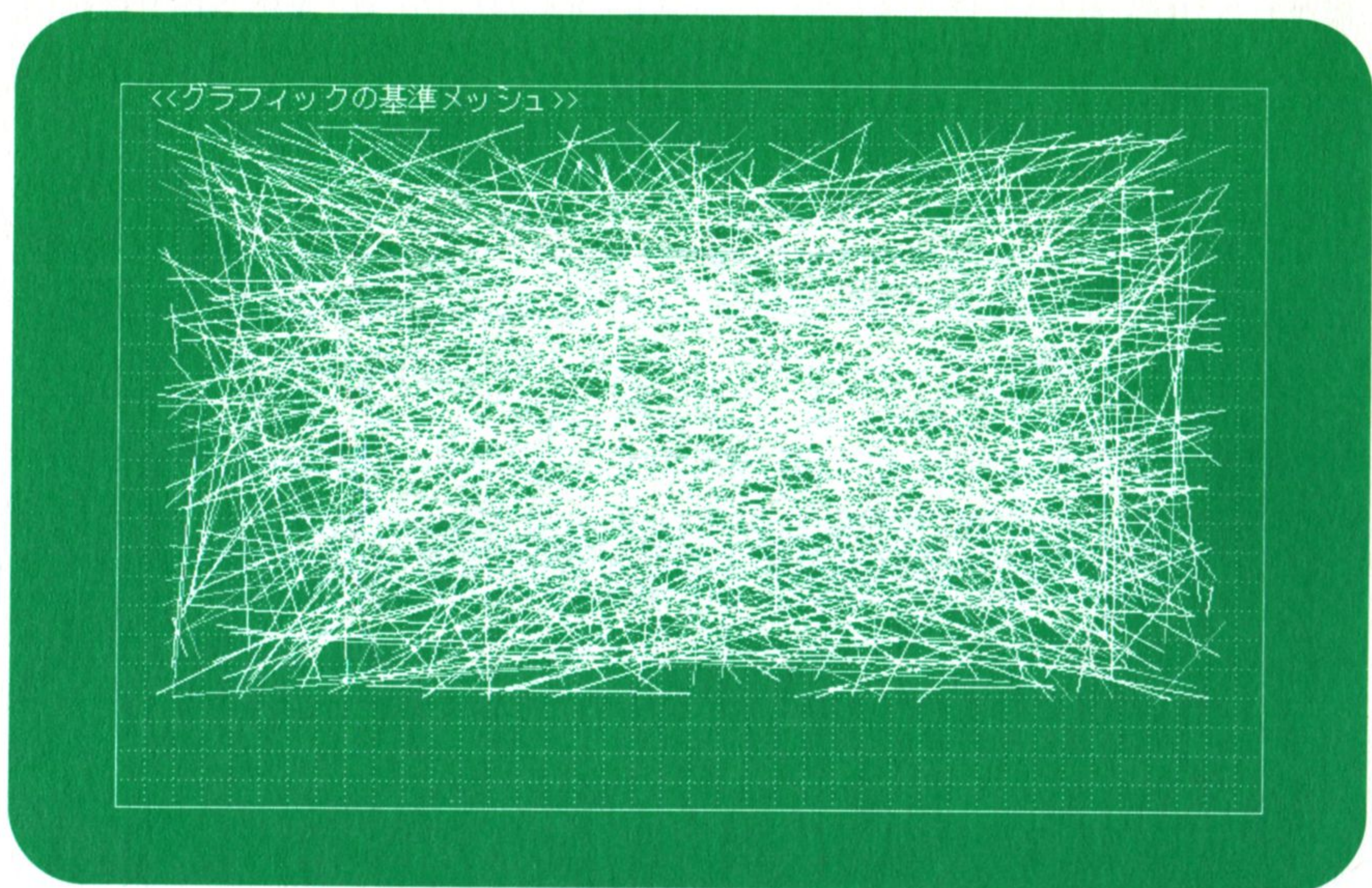
<<グラフィックの基準メッシュ>>

何かキーを押してください

2　線の乱数

このプログラムもグラフィックの基準メッシュは省略できます。また、乱数で発生するX軸の始点は、20～620までです。終点のX軸、始点、終点のY軸はどうなっているか、プログラムを見てください。ちなみに、X1がX軸の始点、Y2がYの終点の乱数の値となります。

```
'GR-PR204.BAS
 RANDOMIZE VAL(RIGHT$(TIME$, 2))
 SCREEN 0
 CLS
    CL = 2
   LINE (0, 0)-(639, 399), 2, B
 FOR X = 0 TO 639 STEP 16
    LINE (X, 0)-(X, 399), CL, , &H1111
 NEXT
 FOR Y = 0 TO 399 STEP 16
    LINE (0, Y)-(639, Y), CL, , &H1111
 NEXT
 LOCATE 1, 1
        PRINT "  <<グラフィックの基準メッシュ>>"
 '線の発生
 CL1 = 7: X3 = 1: Y3 = 1
 FOR I = 0 TO 1000
 CL2 = I MOD 6 + 1
        X1 = INT(RND(1) * 600) + 20
        Y1 = INT(RND(1) * 320) + 20
        X2 = INT(RND(1) * 600) + 20
        Y2 = INT(RND(1) * 320) + 20
     LINE (X1, Y1)-(X2, Y2), CL1, B
 NEXT
 END
```

<<グラフィックの基準メッシュ>>

2.3　三角形の作成

三角形は、直線を3つ使って作ります。当然、4つだと四角形、5つだと五角形と、多角形を作る原点が三角形にあります。そして、この多角形の原点の作り方は、

1．始点、終点を指定したLINE文を3つ用意する方法

2．始点を決めたあと線を延ばしていく方法

3．指定する位置をデータとして取り込んでいく方法

の3つが考えられます。どの方法でも結果は同一ですから、自分にあった方法を見つけてください。

1　LINE文を使った、2つの三角形

画面位置を指定するグラフィックスのメッシュの上に三角形が描かれます。三角形を描くだけであれば、メッシュの部分は省略できます。

```
'GR-PR205.BAS
 SCREEN 0
 CLS
   CL = 2
  LINE (0, 0)-(639, 399), 2, B
FOR X = 0 TO 639 STEP 16
   LINE (X, 0)-(X, 399), CL, , &H1111
NEXT
FOR Y = 0 TO 399 STEP 16
   LINE (0, Y)-(639, Y), CL, , &H1111
NEXT
LOCATE 1, 1
        PRINT "  <<グラフィックの基準メッシュ>>"
'三角形を2個　LINE文
        LINE (200, 50)-(50, 200), 5
        LINE (50, 200)-(300, 300), 5
        LINE (300, 300)-(200, 50), 5

        LINE (300, 20)-(150, 360), 7
        LINE (150, 360)-(500, 300), 7
        LINE (500, 300)-(300, 20), 7
END
```

2 始点を決めて線を延ばす、2つの三角形

始点（PSET）のX軸、Y軸の値を決めたあと、線を延ばしたい位置のX軸、Y軸の値を入力します。三角形の最後は、始点に戻ってきます。

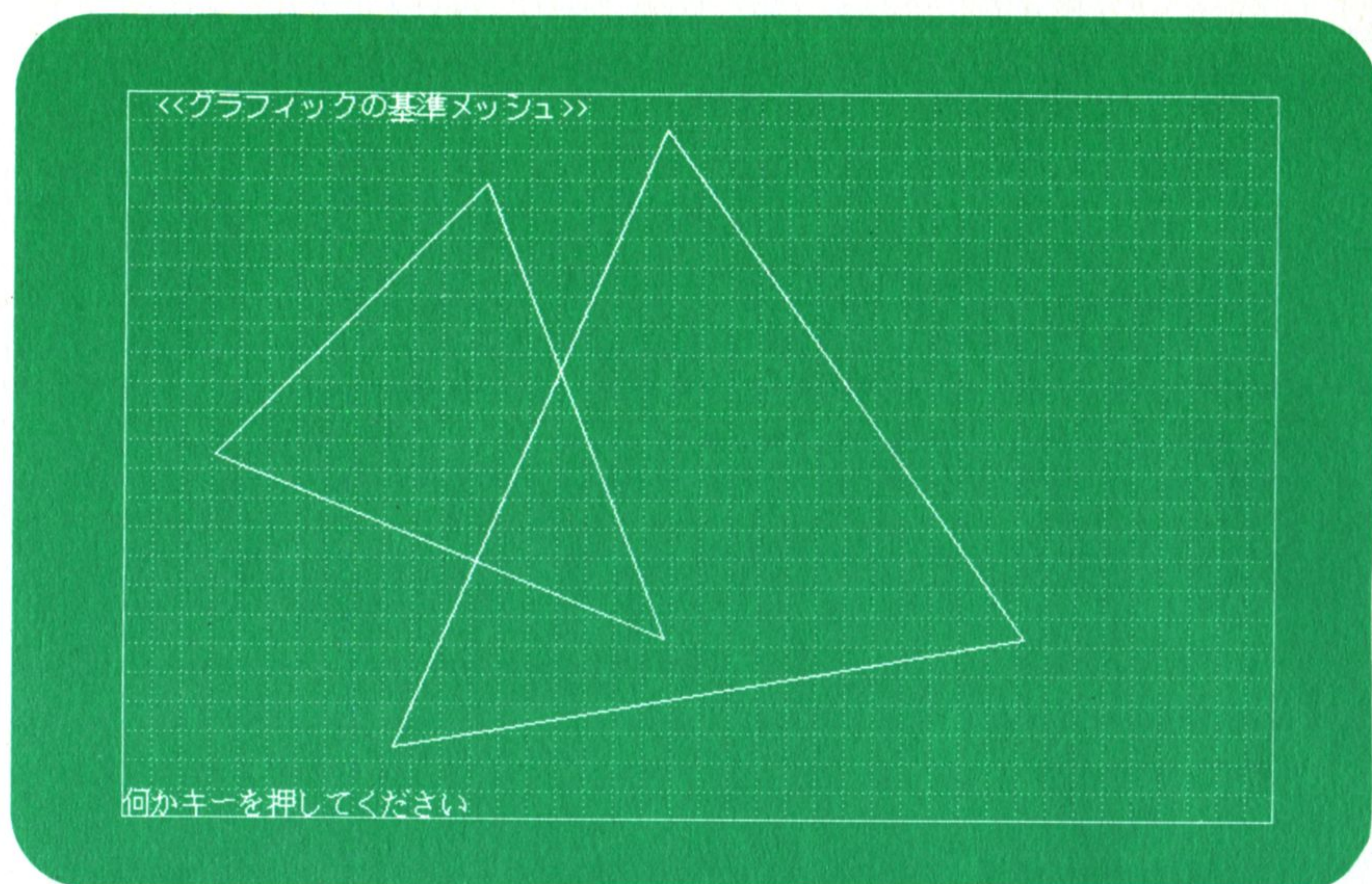
```
'GR-PR206.BAS
 SCREEN 0
 CLS
    CL = 2
   LINE (0, 0)-(639, 399), 2, B
 FOR X = 0 TO 639 STEP 16
    LINE (X, 0)-(X, 399), CL, , &H1111
 NEXT
 FOR Y = 0 TO 399 STEP 16
    LINE (0, Y)-(639, Y), CL, , &H1111
 NEXT
 LOCATE 1, 1
        PRINT "  <<グラフィックの基準メッシュ>>"
'三角形を２個  LINE文
        PSET (200, 50), 5
        LINE -(50, 200), 5
        LINE -(300, 300), 5
        LINE -(200, 50), 5

        PSET (300, 20), 7
        LINE -(150, 360), 7
        LINE -(500, 300), 7
        LINE -(300, 20), 7
 END
```

3　DATA文を使った2つの三角形

データは、X軸の値、Y軸の値、線の色の3つを一組にしています。READされるDATAは同じ数でなくてはなりません。グラフィックスのメッシュは省略できます。

```
'GR-PR207.BAS
 SCREEN 0
 CLS
    CL = 2
   LINE (0, 0)-(639, 399), 2, B
 FOR X = 0 TO 639 STEP 16
    LINE (X, 0)-(X, 399), CL, , &H1111
 NEXT
 FOR Y = 0 TO 399 STEP 16
    LINE (0, Y)-(639, Y), CL, , &H1111
 NEXT
 LOCATE 1, 1
        PRINT "  <<グラフィックの基準メッシュ>>"
 '三角形を2個  READ...DATA文
   FOR I = 1 TO 8
        READ X1, Y1, CL
      IF I = 1 OR I = 5 THEN
           PSET (X1, Y1), CL
      ELSE
           LINE -(X1, Y1), CL
      END IF
   NEXT
   DATA 200,50,5,50,200,5,300,300,5,200,50,5
   DATA 300,20,7,150,360,7,500,300,7,300,20,7
 END
```

<<グラフィックの基準メッシュ>>

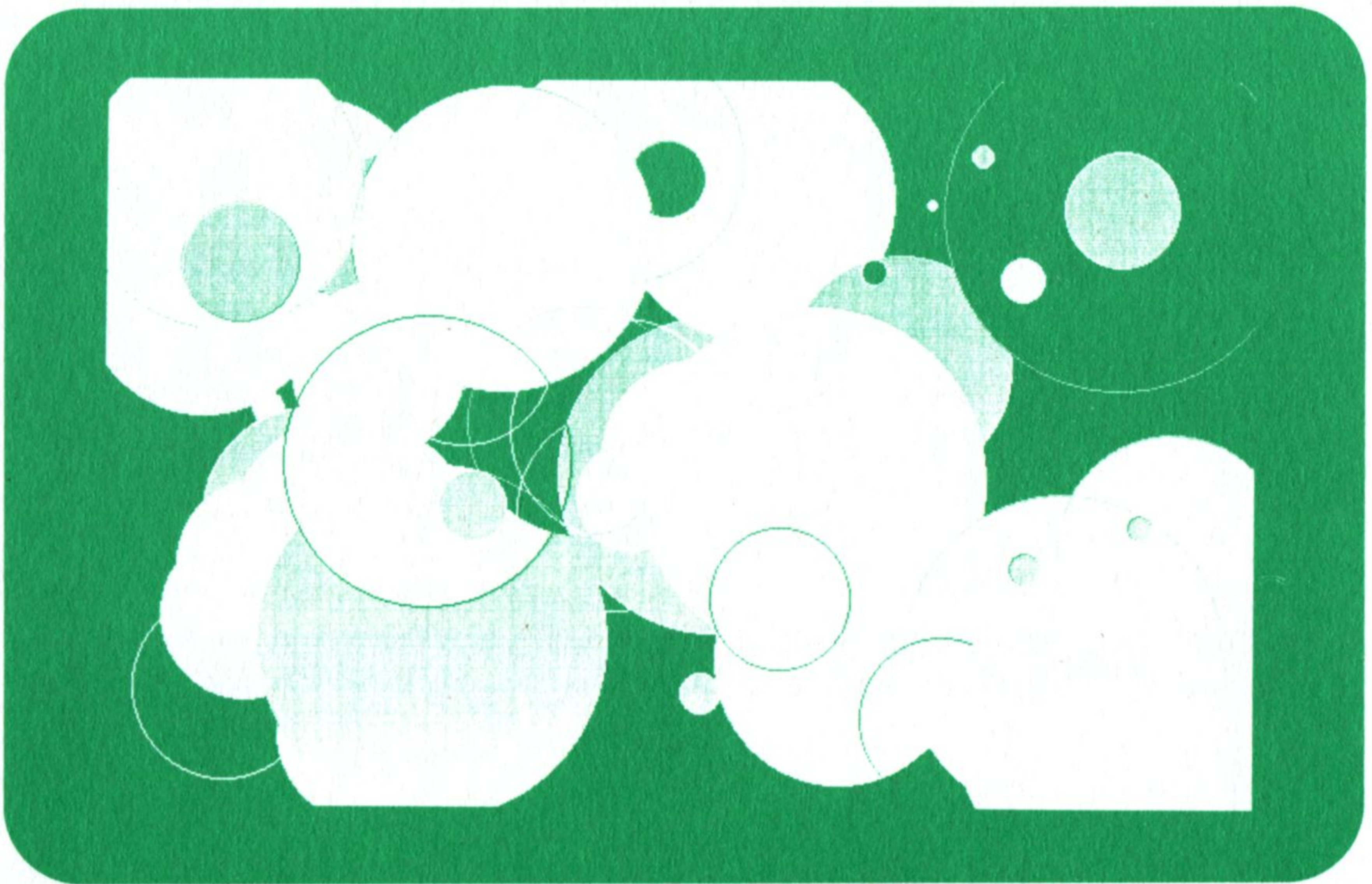
何かキーを押してください

2.4 乱数を使った円と矩形

円は中心点（Y,X）、矩形は、始点(X1,Y1）と終点（X2,Y2）を乱数で作ります。線の色（CL2）も乱数です。大・中・小と円と矩形を黒～白までの七色で塗りつぶしていきます。

1 乱数を使った100個の円

乱数を使った円を作ります。円の中心点はX1とY1。半径はR。色は、CL1とCL2を使います。FOR ...NEXTを使って、100個の円を作っています。

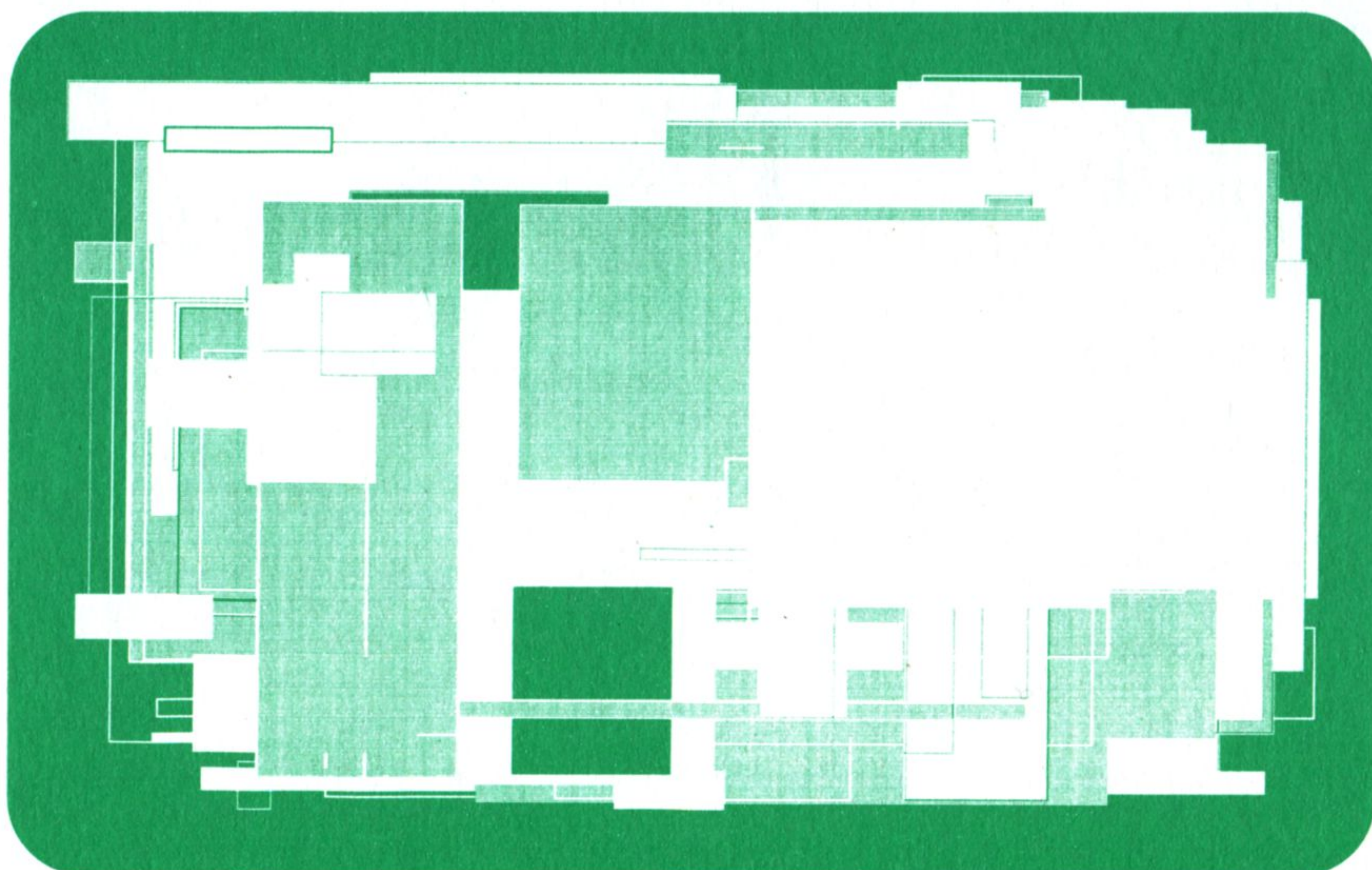
```
'GR-PR208.BAS
 RANDOMIZE VAL(RIGHT$(TIME$, 2))
SCREEN 0
CLS
```

```
FOR I = 1 TO 100
        X1 = INT(RND(1) * 590) + 20
        Y1 = INT(RND(1) * 350) + 20
        R = INT(RND(1) * 100)
        CL1 = INT(RND(1) * 8)
        CL2 = INT(RND(1) * 8)

            CIRCLE (X1, Y1), R, CL2
            PAINT (X1, Y1), CL1, CL2
NEXT
```

＜試してみよう＞

1．FOR I=1 TO 100 の数字100を変えてみてください。作られる円の数が変化します。
2．変数CL2を数字1～7までのどれかに置き換えてみてください。画面の印象がどのように変わりましたか。

2 乱数を使った四角形

変数、X1とY1は始点で、変数X2,Y2は終点です。PAINT文を使った塗りつぶしは、円のときのように中心点を出すことができませんのでちょっとした工夫が必要です。(X1+X2)/2と(Y1+Y2)/2を使っています。なぜでしょう。考えてみてください。

```
'GR-PR209.BAS
 RANDOMIZE VAL(RIGHT$(TIME$, 2))
SCREEN 0
CLS

FOR I = 1 TO 100
        X1 = INT(RND(1) * 590) + 20
        Y1 = INT(RND(1) * 350) + 20
        X2 = INT(RND(1) * 590) + 20
        Y2 = INT(RND(1) * 350) + 20
        CL1 = INT(RND(1) * 8)
        CL2 = INT(RND(1) * 8)
            LINE (X1, Y1)-(X2, Y2), CL2, B
       PAINT ((X1 + X2) / 2, (Y1 + Y2) / 2), CL1, CL2
NEXT
```

＜試してみよう＞

変数H=RND(1)*3の数字3を1に変えてみてください。

```
'GR-PR210.BAS
 RANDOMIZE VAL(RIGHT$(TIME$, 2))
SCREEN 0:CLS
FOR I = 1 TO 100
        X1 = INT(RND(1) * 590) + 20:Y1 = INT(RND(1) * 350) + 20
        R = INT(RND(1) * 100):H =RND(1)*3
        CL1 = INT(RND(1) * 8):CL2 = INT(RND(1) * 8)
            CIRCLE (X1, Y1), R, CL2,,,H
            PAINT (X1, Y1), CL1, CL2
NEXT
```

3　乱数を使った色の発生

色の発生の方法には、0から7までの黒を含むカラーコードの指定の仕方と、1から7までの黒を含まないカラーコードの発生の方法があります。

黒を含むカラーコードの発生　　CL=INT(RND(1)*8)
黒を含まないカラーコードの発生　CL=INT(RND(1)*7)+1

このプログラムは、0～7までの数値を乱数で発生させたときの分布状態を表示しています。乱数を発生させる回数をI、表示をPSETで行なっています。発生したコードは、計算してさらに%表示を行なっています。

```
PRINT INT(CL(J - 1) / I * 1000 + .5) / 10; "%"
```

加算したそれぞれのコードCL(J-1)を全体の数字Iで割ったあと、1000倍して、四捨五入を行なうために0.5を加えて、下一桁までを表示するため10で割ります。

```
'GR-PR211.BAS
 RANDOMIZE VAL(RIGHT$(TIME$, 2))
SCREEN 0: CLS
 XX = 50: YY = 2
 FOR J = 1 TO 8
    READ TI$: LOCATE (J * 2), 12: PRINT TI$
 NEXT
    DATA "黒","青","緑","水","赤","紫","黄","白"
    W = 10
 FOR I = 1 TO 40000             'I/W=<4000で設定
        CL = INT(RND(1) * 8): B = CL + 1
      IF CL = 0 THEN
        CL(0) = CL(0) + 1: PSET (XX + CL(0) / W, YY * B * 16 + 1), 7
      ELSEIF CL = 1 THEN
        CL(1) = CL(1) + 1: PSET (XX + CL(1) / W, YY * B * 16 + 1), 1
      ELSEIF CL = 2 THEN
        CL(2) = CL(2) + 1: PSET (XX + CL(2) / W, YY * B * 16 + 1), 2
      ELSEIF CL = 3 THEN
        CL(3) = CL(3) + 1: PSET (XX + CL(3) / W, YY * B * 16 + 1), 3
      ELSEIF CL = 4 THEN
        CL(4) = CL(4) + 1: PSET (XX + CL(4) / W, YY * B * 16 + 1), 4
      ELSEIF CL = 5 THEN
        CL(5) = CL(5) + 1: PSET (XX + CL(5) / W, YY * B * 16 + 1), 5
      ELSEIF CL = 6 THEN
        CL(6) = CL(6) + 1: PSET (XX + CL(6) / W, YY * B * 16 + 1), 6
      ELSEIF CL = 7 THEN
        CL(7) = CL(7) + 1: PSET (XX + CL(7) / W, YY * B * 16 + 1), 7
      END IF
NEXT
 FOR J = 1 TO 8
   LOCATE (J * 2), 14
    PRINT CL(J - 1), INT(CL(J - 1) / I * 1000 + .5) / 10; "%"
 NEXT
```

黒 5010	12.5 %
青 5021	12.6 %
緑 5056	12.6 %
水 5105	12.8 %
赤 4986	12.5 %
紫 4967	12.4 %
黄 4955	12.4 %
白 4900	12.2 %

＜試してみよう＞

1．乱数原数値です。100回分画面に表示します。

```
FOR I=1 TO 100
PRINT RND(1),
NEXT
```

2．原数値を10にして、丸めた（INT）あと、画面に100回分表示します。

```
FOR I=1 TO 100
PRINT RND(1),
NEXT
```

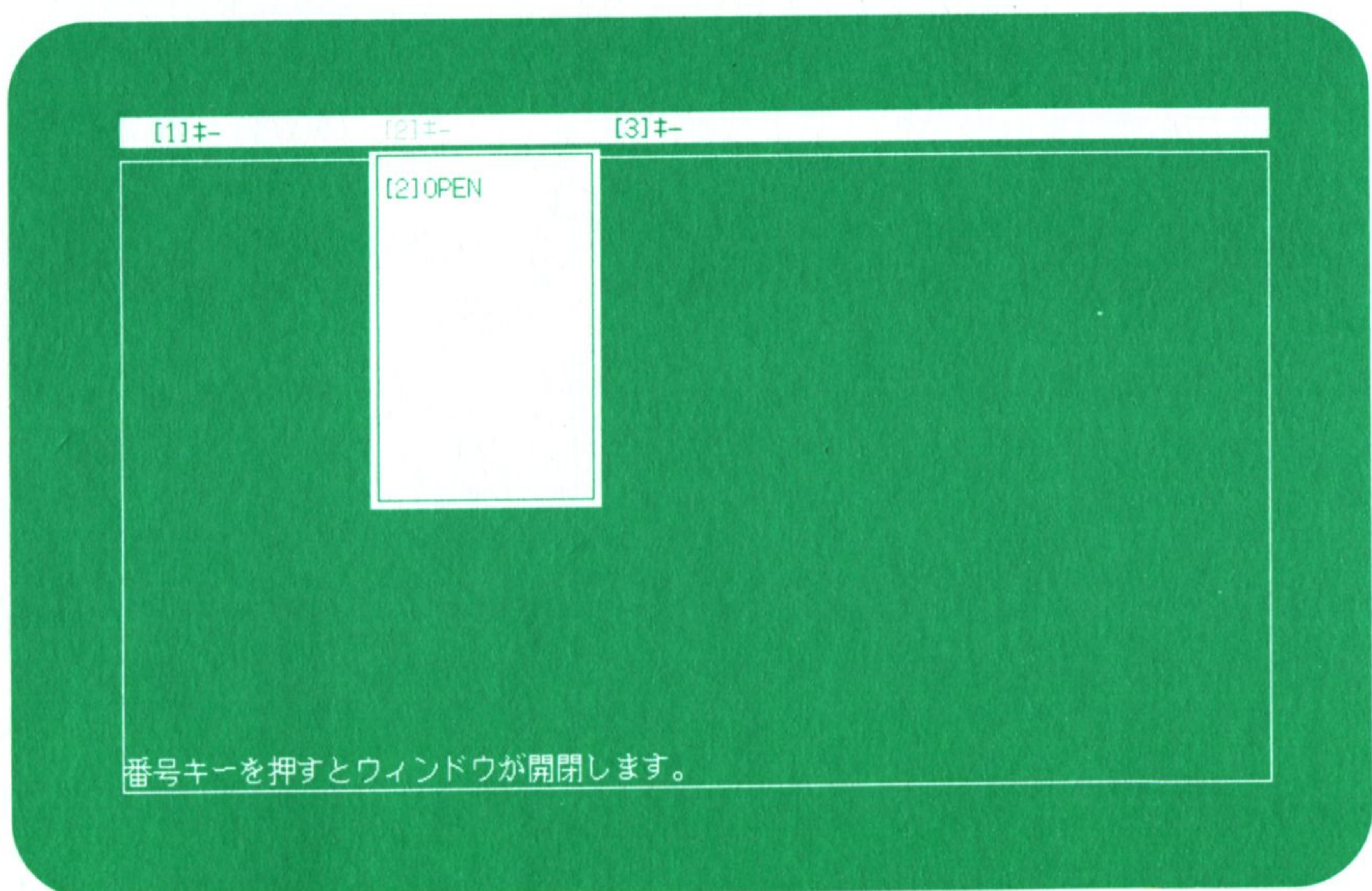
2.5　PUT、GETでウィンドウを作る

Quick BASICを起動してプログラムを編集するとき、サブメニューのウィンドウをオープンします。直線と四角形の応用例として、ウィンドウのプログラムを紹介します。

画面の状態をGETで読み込み、ウィンドウをLINE文で表示したあと、GET文で

読み込んだ画面データを上塗りする感じでウィンドウを消します。画面は黒地になっていますが、PAINT文で中間色が出せることは学びました。ぜひ、自分のウィンドウを完成させてください。

```
'GR-PR012 PUT&GETを使ったウィンドウ
DIM W%(9460): OCF = 1 'DIM=((128+7)/8)*197*7)
 SCREEN 0: CLS
  LINE (0, 0)-(639, 16), 3, BF
  LINE (0, 24)-(639, 371), 7, B
   COLOR 0
    LOCATE 1, 3: PRINT "[1]ｷｰ"
     LOCATE 1, 19: PRINT "[2]ｷｰ"
    LOCATE 1, 35: PRINT "[3]ｷｰ "
   LOCATE 23, 1: COLOR 7: PRINT "番号キーを押すとウィンドウが開閉します。";
    GET (10, 20)-(138, 216), W%
DO
 WHILE OCF = 1
  DO
   AN$ = INKEY$
    IF AN$ = "1" OR AN$ = "2" OR AN$ = "3" THEN
     OCF = 1: GOSUB OWIN:
    END IF
     AN$ = ""
  LOOP UNTIL AN$ = ""
 WEND
  DO
   ANS$ = INKEY$
    IF VAL(ANS$) = FL THEN
      OCF = 1
      COLOR 0
      LOCATE 1, (16 * FL) - 13: PRINT "["; ANS$; "]ｷｰ";
      COLOR 7
      PUT (10 + (FL * 128) - 128, 20), W%, PSET
    END IF
```

```
        ANS$ = ""
    LOOP UNTIL ANS$ = ""
   LOOP
     END

OWIN:
   FL = VAL(AN$): COLOR 4
      LOCATE 1, (16 * FL) - 13: PRINT "["; AN$; "]キー";
     COLOR 0
    LINE ((10 + FL * 128) - 128, 20)-((138 + FL * 128) - 128, 216), 7, BF
    LINE ((14 + FL * 128) - 128, 22)-((134 + FL * 128) - 128, 212), 0, B
     LOCATE 3, (16 * FL) - 13: PRINT "["; AN$; "]OPEN";
      LOCATE 3, (16 * FL) - 13: PRINT "["; AN$; "]OPEN";
       OCF = 2
RETURN
```

[1]キー　[2]キー　[3]キー

[2]OPEN

番号キーを押すとウィンドウが開閉します。

第3章　曲　線

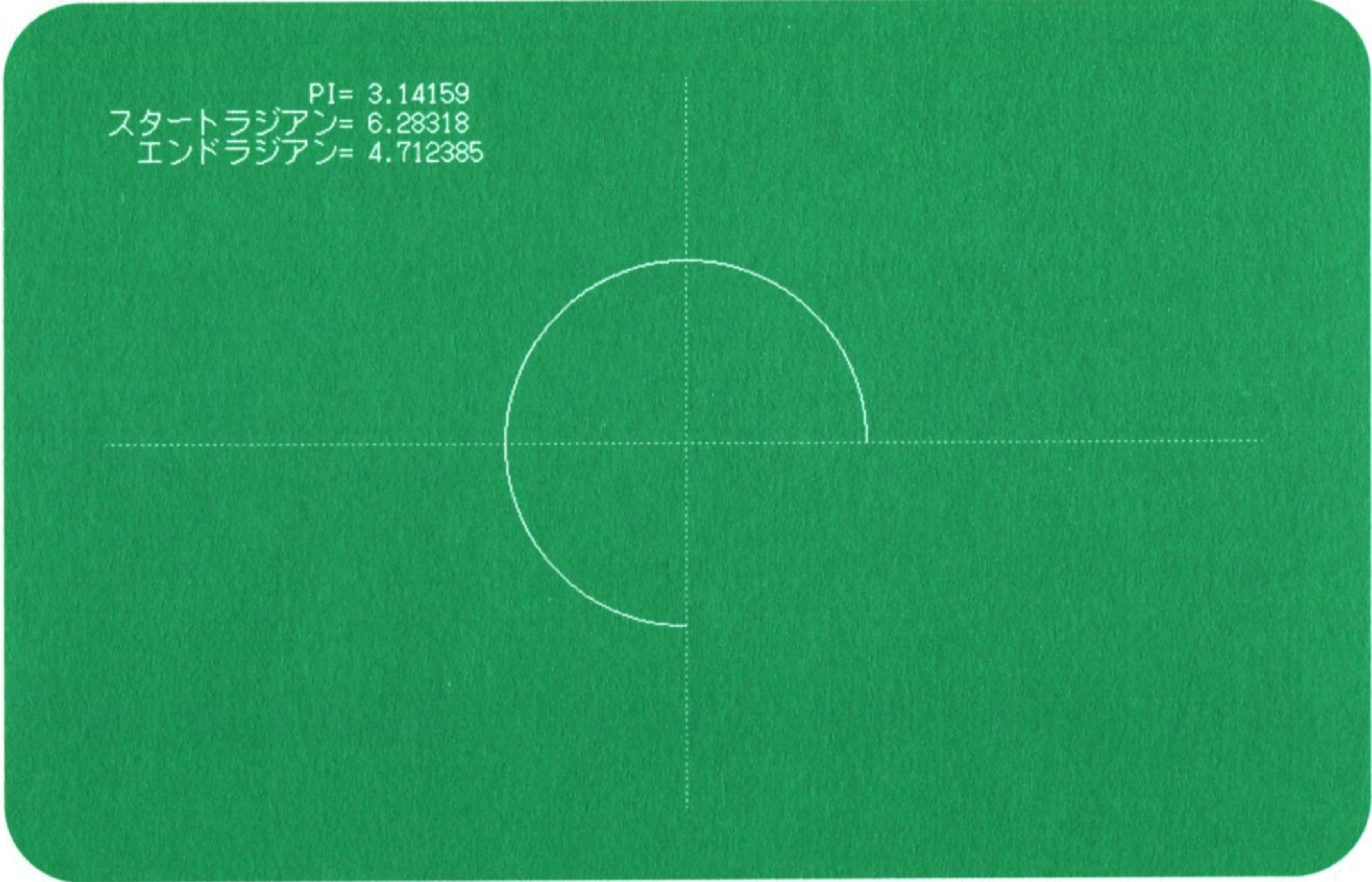
3.1 CIRCLE文を使った円弧

円を描くCIRCLE文にオプションをつけることで、円弧を描くことができます。

```
CIRCLE(X,Y),R,CL,S1,E1,TU
```

X,Yは円（円弧）の中心座標。Rは半径。PC-9801の画面はY軸方向が400ですから、Y軸座標200として、R（半径）は150位がわかりやすいと思います。

CLは円（円弧）の色です。S1はスタート角度。E1はエンド角度を表わし、円を描く開始位置は、時計の3時の位置の0ラジアンから始まり、時計と反対回りの12時（角度90度）が、0.5πラジアン、9時（180度）が1πラジアンとなります。0ラジアンは、2πラジアンと同じです。最後のTUは、楕円を作る比率です。

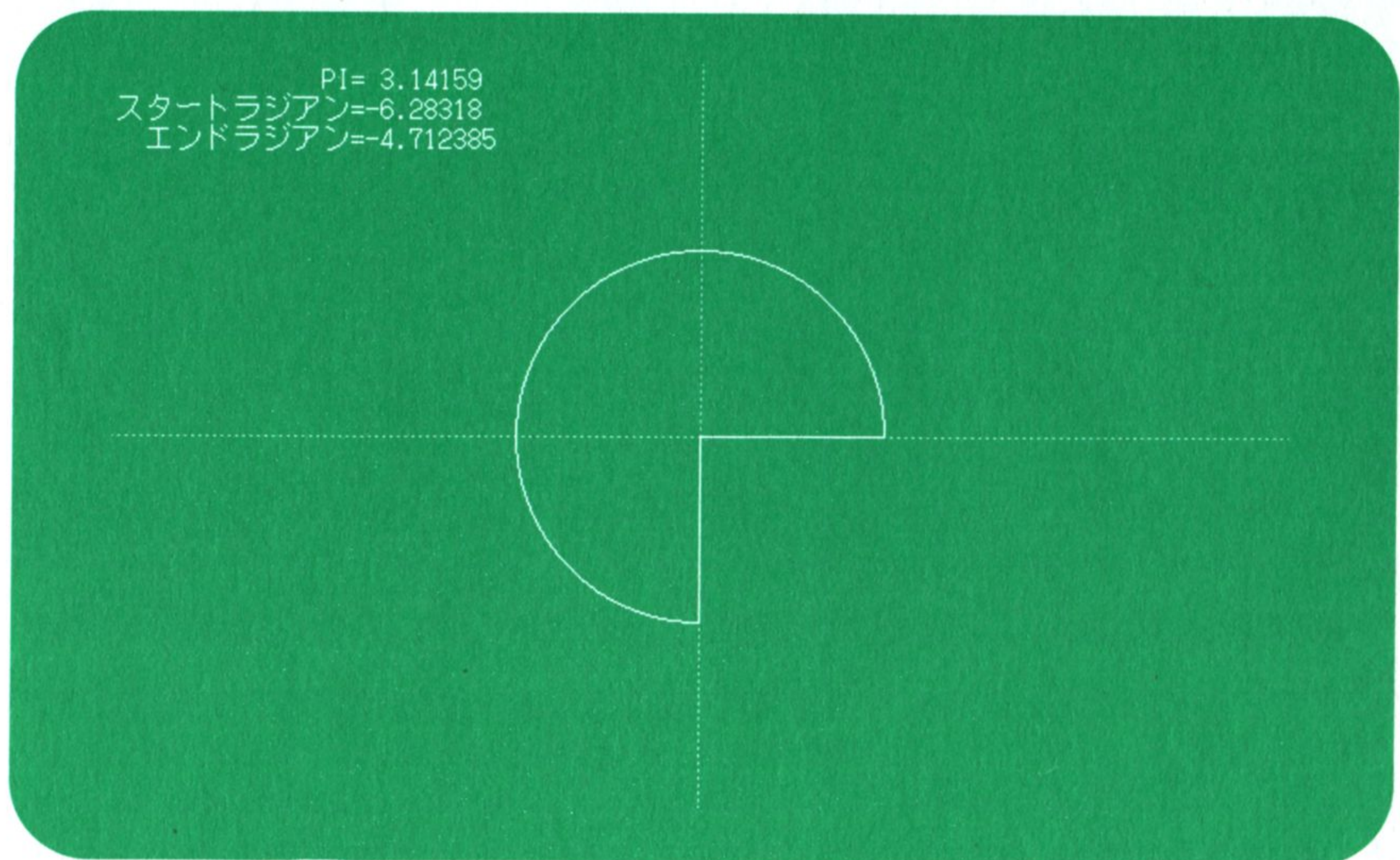
```
'GR-PR301
SCREEN 0
CLS
    LINE (320, 0)-(320, 400), 2, , &H9999
    LINE (0, 200)-(640, 200), 2, , &H9999
```

```
PI = 3.14159
S1 = 2 * PI
E1 = 1.5 * PI
    LOCATE 1, 1
    PRINT "              PI="; PI
    PRINT "スタートラジアン="; S1
    PRINT "  エンドラジアン="; E1
CIRCLE (320, 200), 100, , S1, E1
```

```
          PI= 3.14159
スタートラジアン= 6.28318
  エンドラジアン= 4.712385
```

1 扇形を描く

CIRCLE文のスタート角度、エンド角度に負の実数を与えると扇形で表示されます。S1とE1に－（マイナス）記号がついていることに注意してください。この扇形のプログラムをいくつか組み合わせることで、円グラフを描くことができます。円弧を作るためには、－記号が必要ですから、0ラジアンは使うことができません。そこで、例題のプログラムは、－2*PIを使っていますが、0*PI－0.0001という使い方があります。

```
'GR-PR302
SCREEN 0
CLS
    LINE (320, 0)-(320, 400), 2, , &H9999
    LINE (0, 200)-(640, 200), 2, , &H9999
        PI = 3.14159
        S1 = -2 * PI
        E1 = -1.5 * PI
            LOCATE 1, 1
            PRINT "              PI="; PI
            PRINT "スタートラジアン="; S1
            PRINT "  エンドラジアン="; E1
        CIRCLE (320, 200), 100, , S1, E1
```

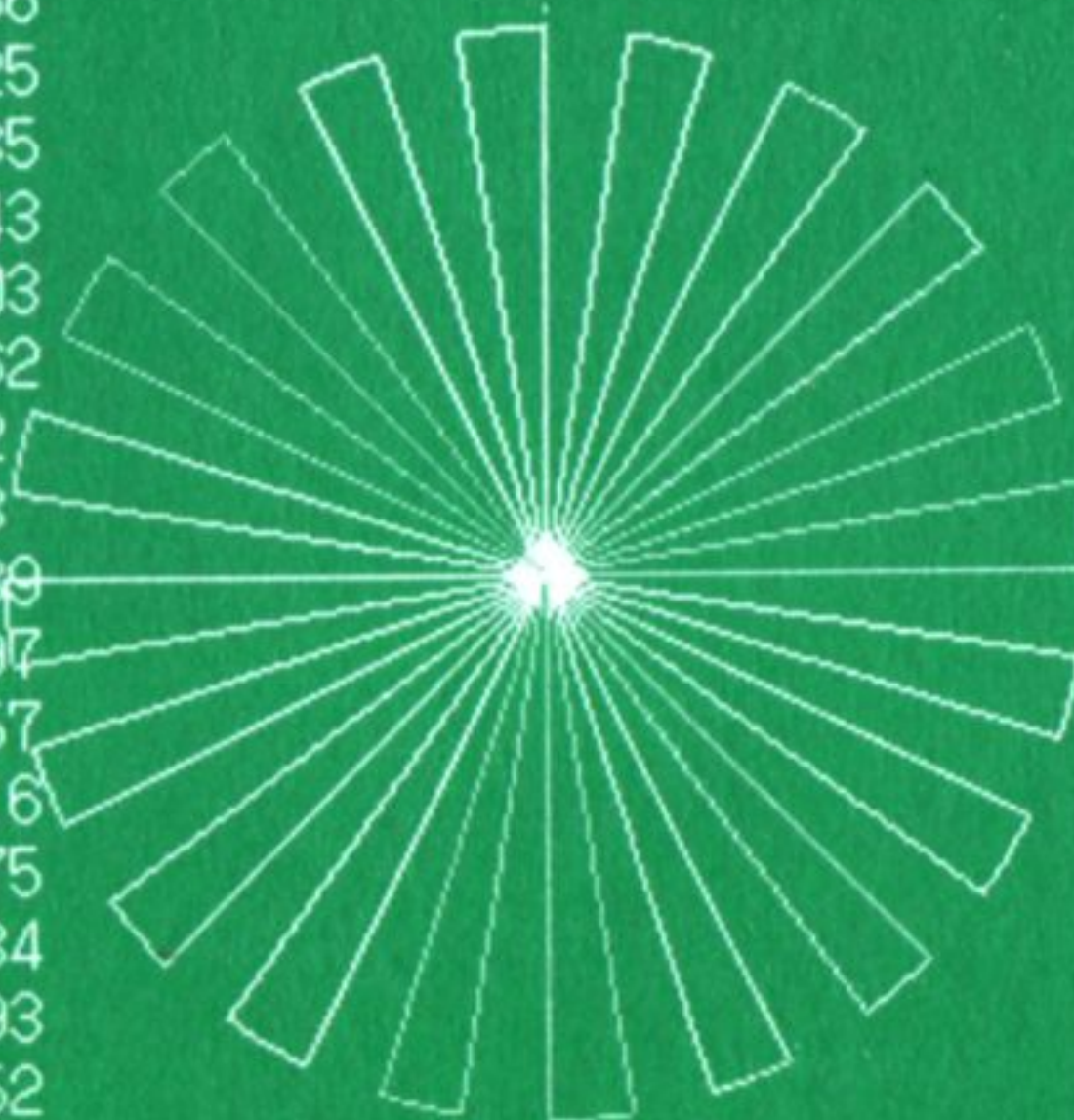
```
              PI= 3.14159
スタートラジアン=-6.28318
  エンドラジアン=-4.712385
```

CIRCLE文を次のように書き換えてみましょう。実行結果は、同じになりました。負の整数を作るために、できるだけ小さな負の数値を作りました。なんとなく、コンピュータをダマしたような気分がしませんか。

```
S1 = 0 * PI-0.0001
E1 = -1.5 * PI
```

2 連続した扇形を描く

スタート、エンドラジアンを負の実数を使うことで、円弧が作れました。今度は、変数KAZの中の数字で扇形の数を変えることができます。ここでは、20個の連続扇形を作ります。

```
'GR-PR303
SCREEN 0
CLS
    LINE (320, 0)-(320, 400), 2, , &H9999
    LINE (0, 200)-(640, 200), 2, , &H9999
        KAZ = 20
        PI = 3.14159
        S1 = -2 * PI / KAZ
        E1 = -2 * PI / (KAZ * 2)
    FOR I = 0 TO KAZ - 1
        PRINT "S2="; S2, "E2="; E2
        CL = I MOD 7 + 1
        S2 = I * S1 - .00001
        E2 = S2 + E1
            CIRCLE (320, 200), 120, CL, S2, E2
    NEXT
```

```
S2= 0           E2= 0
S2=-.00001      E2=-.1570895
S2=-.314169     E2=-.4712485
S2=-.628328     E2=-.7854075
S2=-.942487     E2=-1.099566
S2=-1.256646    E2=-1.413725
S2=-1.570805    E2=-1.727885
S2=-1.884964    E2=-2.042043
S2=-2.199123    E2=-2.356203
S2=-2.513282    E2=-2.670362
S2=-2.827441    E2=-2.98452
S2=-3.1416      E2=-3.29868
S2=-3.455759    E2=-3.612839
S2=-3.769918    E2=-3.926997
S2=-4.084077    E2=-4.241157
S2=-4.398236    E2=-4.555316
S2=-4.712395    E2=-4.869475
S2=-5.026554    E2=-5.183634
S2=-5.340713    E2=-5.497793
S2=-5.654872    E2=-5.811952
```

<試してみよう>

P.173の最後のプログラムリストに1/2を追加して、CIRCLE(320,200),120,CL,S2,E2,1/2と、書き換えてみてください。真円が楕円に変わります。

3.2 関数を使った曲線

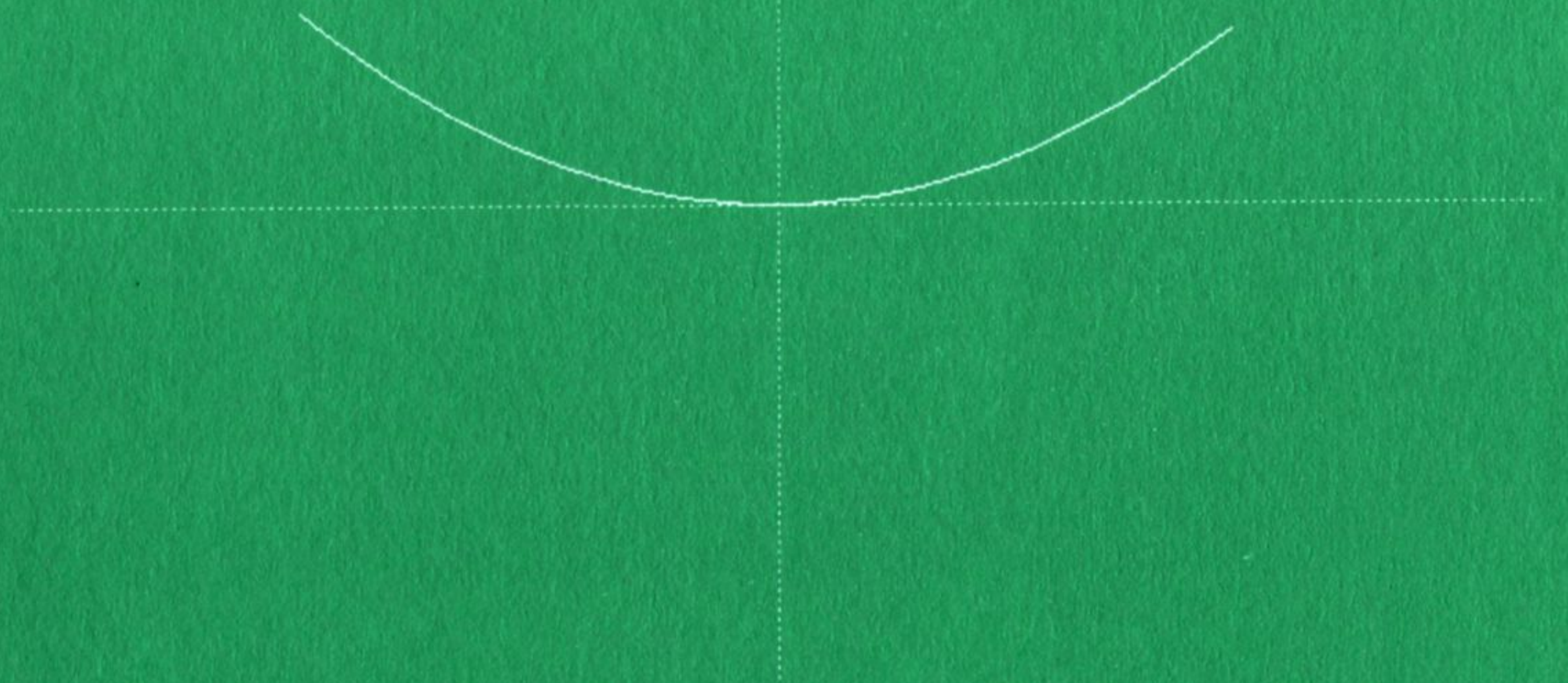
二次関数Y=aX^2、および三次関数Y=aX^3を使って、曲線を描くことができます。学校で学んだことを実際に目で見ることができます。変数X2とY2の数字をいろいろと変えて試してください。

1 二次関数の曲線

INPUTの数値の目安を変数X2とY2に入れています。このプログラムも二次関数の基本的なプログラム例です。

```
'GR-PR304  X=Y^2
SCREEN 0
CLS
FOR J = 1 TO 10
    PRINT "Xの値が9999だったらプログラムを終了します"
    INPUT "関数Y=X^2のXの値を入力してください="; X2
    IF X2 = 9999 THEN END
    INPUT "関数Y=X^2のYの値を入力してください="; Y2
CLS
    LINE (320, 0)-(320, 400), 2, , &H9999
    LINE (0, 200)-(640, 200), 2, , &H9999
        X1 = 320: Y1 = 200
       'X2 =200: Y2 = 100
        XMI = -2: XMA = 2
```

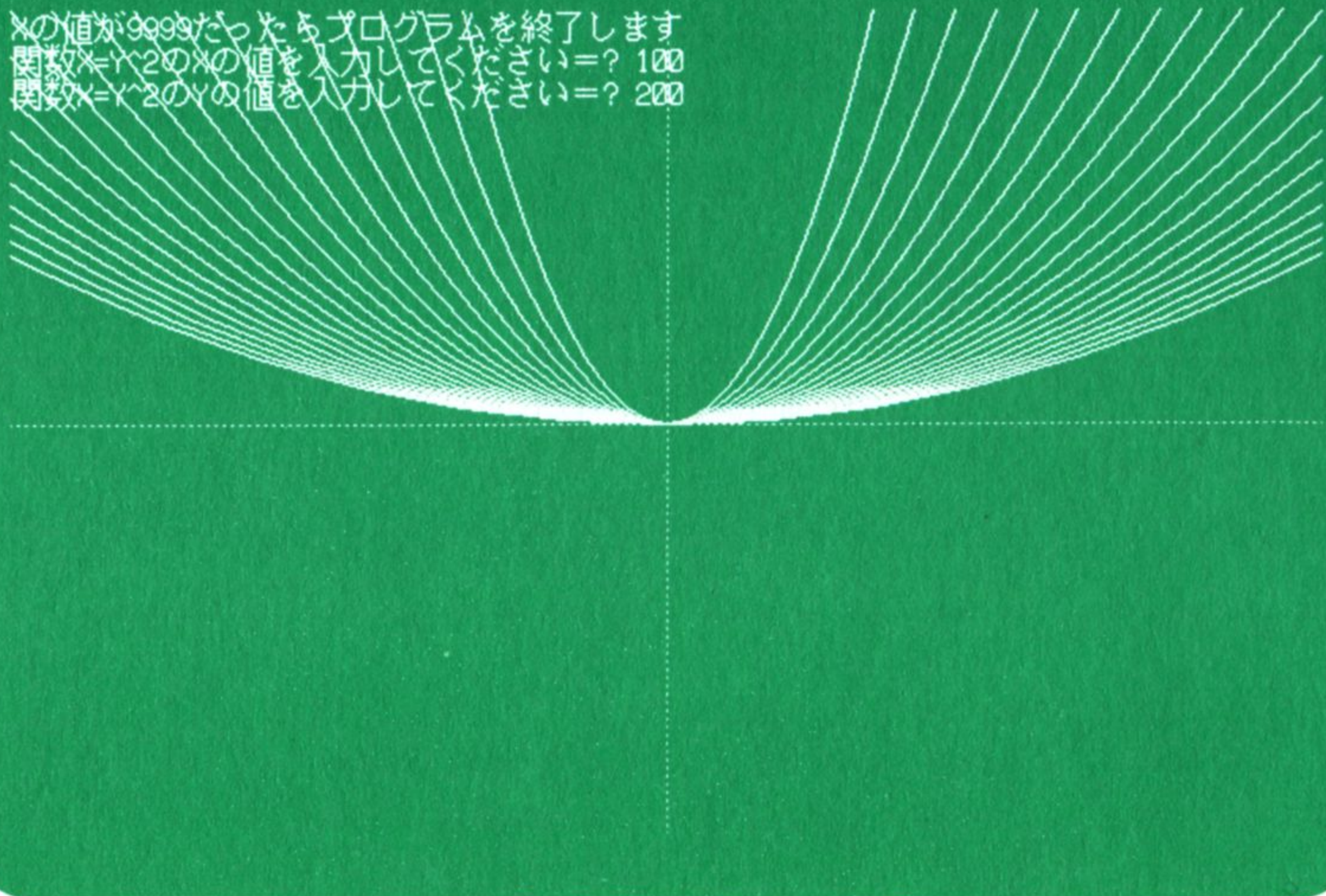
```
FOR X = XMI TO XMA STEP .1
       XX = X1 + X * X2
       yy = Y1 - ((X ^ 2) * Y2)
       IF X = XMI THEN
       PSET (XX, yy), 7
       ELSE
       LINE -(XX, yy)
       END IF
 NEXT
NEXT
```

Xの値が9999だったらプログラムを終了します
関数X=Y^2のXの値を入力してください=?

2　二次関数の応用

適当な数値を入力してください。いろいろな数値で10回画面を楽しむことができます。

```
'GR-PR305  Y=X^2 X移動
SCREEN 0
CLS
FOR J = 1 TO 10
    PRINT "Xの値が9999だったらプログラムを終了します"
    INPUT "関数X=Y^2のXの値を入力してください="; X2
```

```
    IF X2 = 9999 THEN END
    INPUT "関数X=Y^2のYの値を入力してください="; Y2

CLS
    LINE (320, 0)-(320, 400), 2, , &H9999
    LINE (0, 200)-(640, 200), 2, , &H9999
        X1 = 320: Y1 = 200
       'X2 =100: Y2 = 200
        XMI = -2: XMA = 2
    FOR I = 1 TO 5 STEP .2
        FOR X = XMI TO XMA STEP .1
            XX = X1 + X * X2 * I
            yy = Y1 - ((X ^ 2) * Y2)
          IF X = XMI THEN
            PSET (XX, yy), 7
            ELSE
                LINE -(XX, yy)
          END IF
        NEXT
    NEXT
NEXT
```

Xの値が9999だったらプログラムを終了します
関数X=Y^2のXの値を入力してください=? 100
関数X=Y^2のYの値を入力してください=? 200

3　三次関数の曲線

Y軸に対して、対称の曲線となります。視覚で数学の公式を再現できるのは、パソコンのよいところです。

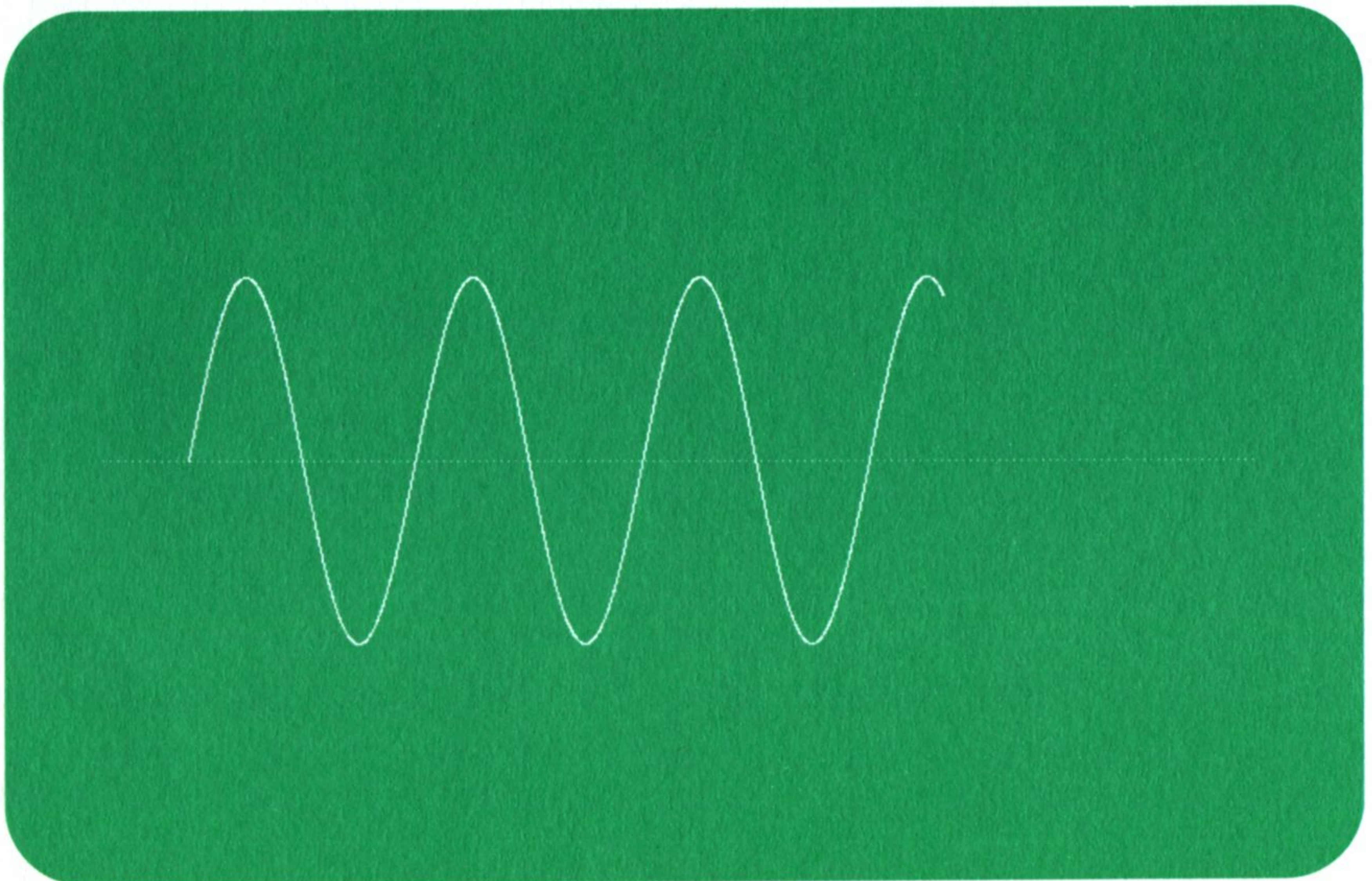
```
'GR-PR306  Y=X^3
SCREEN 0
CLS
FOR J = 1 TO 10
    PRINT "Xの値が9999だったらプログラムを終了します"
    INPUT "関数Y=X^3のXの値を入力してください="; X2
    IF X2 = 9999 THEN END
    INPUT "関数Y=X^3のYの値を入力してください="; Y2

CLS
    LINE (320, 0)-(320, 400), 2, , &H9999
    LINE (0, 200)-(640, 200), 2, , &H9999
        X1 = 320: Y1 = 200
       'X2 =200: Y2 = 100
        XMI = -2: XMA = 2

 FOR X = XMI TO XMA STEP .01
        XX = X1 + X * X2
        yy = Y1 - ((X ^ 3) * Y2)
        IF X = XMI THEN
        PSET (XX, yy), 7
        ELSE
        LINE -(XX, yy)
        END IF
 NEXT
NEXT
```

3.3　三角関数を使う

パソコンの出始めのころは、CIRCLE文がなくて、SINとCOSの三角関数を使って円を作ったものです。そして、サインカーブとコサインカーブを使って、バイオリズムを作ってみるというのが、犬の卒倒（ワンパターン）でした。あなたは、どのようなことに挑戦してみますか。

1　サインカーブ

単純にSINカーブを作ります。変数XMAの数値が、SINカーブの回数です。プログラムでは、7つの山を作ります。

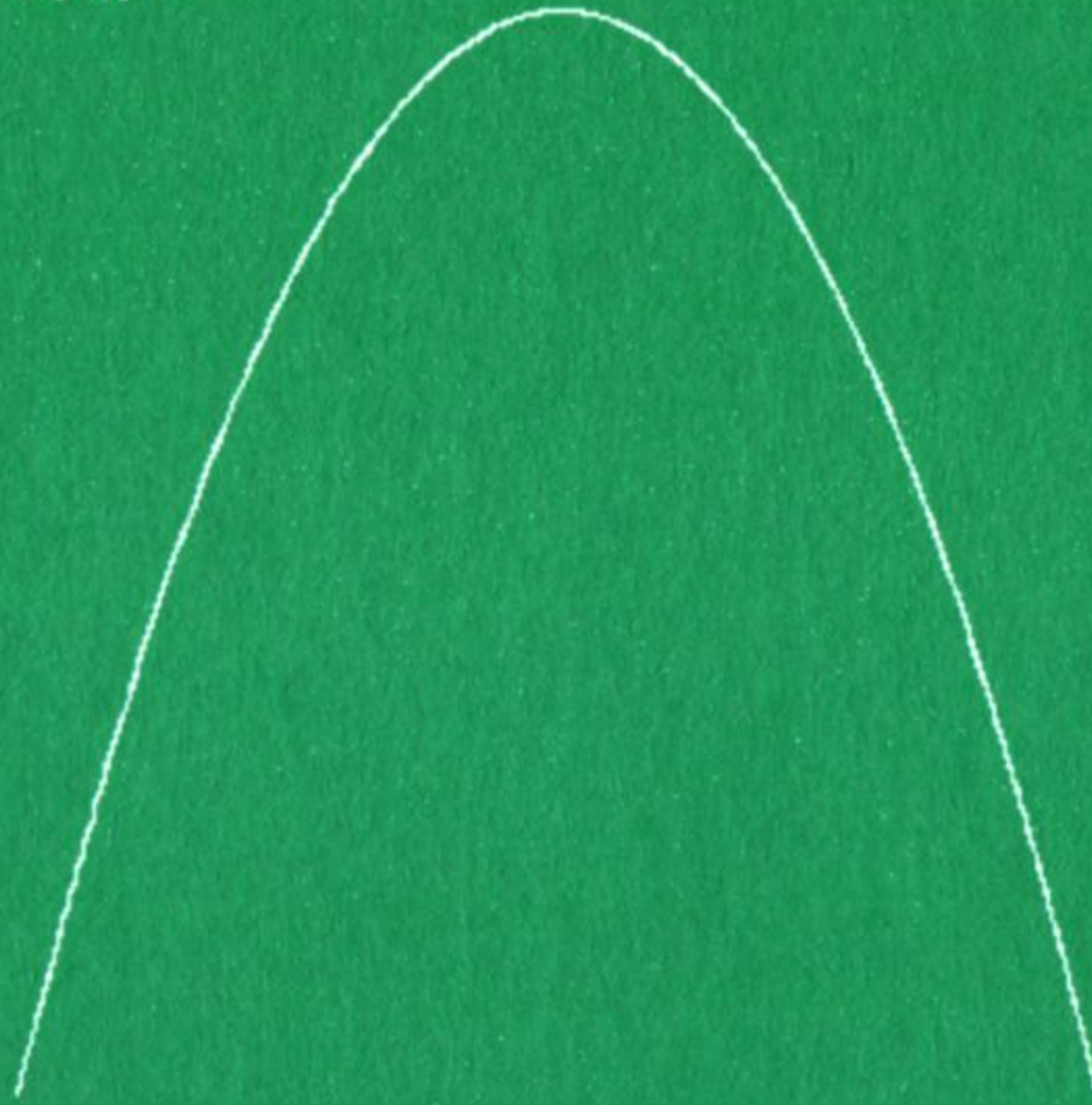
```
'GR-PR307  SIN CURVE
SCREEN 0
CLS
   LINE (0, 200)-(640, 200), 2, , &H8888
       X1 = 50: Y1 = 200
       PI = 3.14159
       X2 = 20: Y2 = 100
       XMI = 0: XMA = 7 * PI
    FOR X = XMI TO XMA STEP .01
            XX = X1 + X2 * X
            yy = Y1 - SIN(X) * Y2
          IF X = XMI THEN
            PSET (XX, yy)
            ELSE
                LINE -(XX, yy)
          END IF
    NEXT
```

＜試してみよう＞

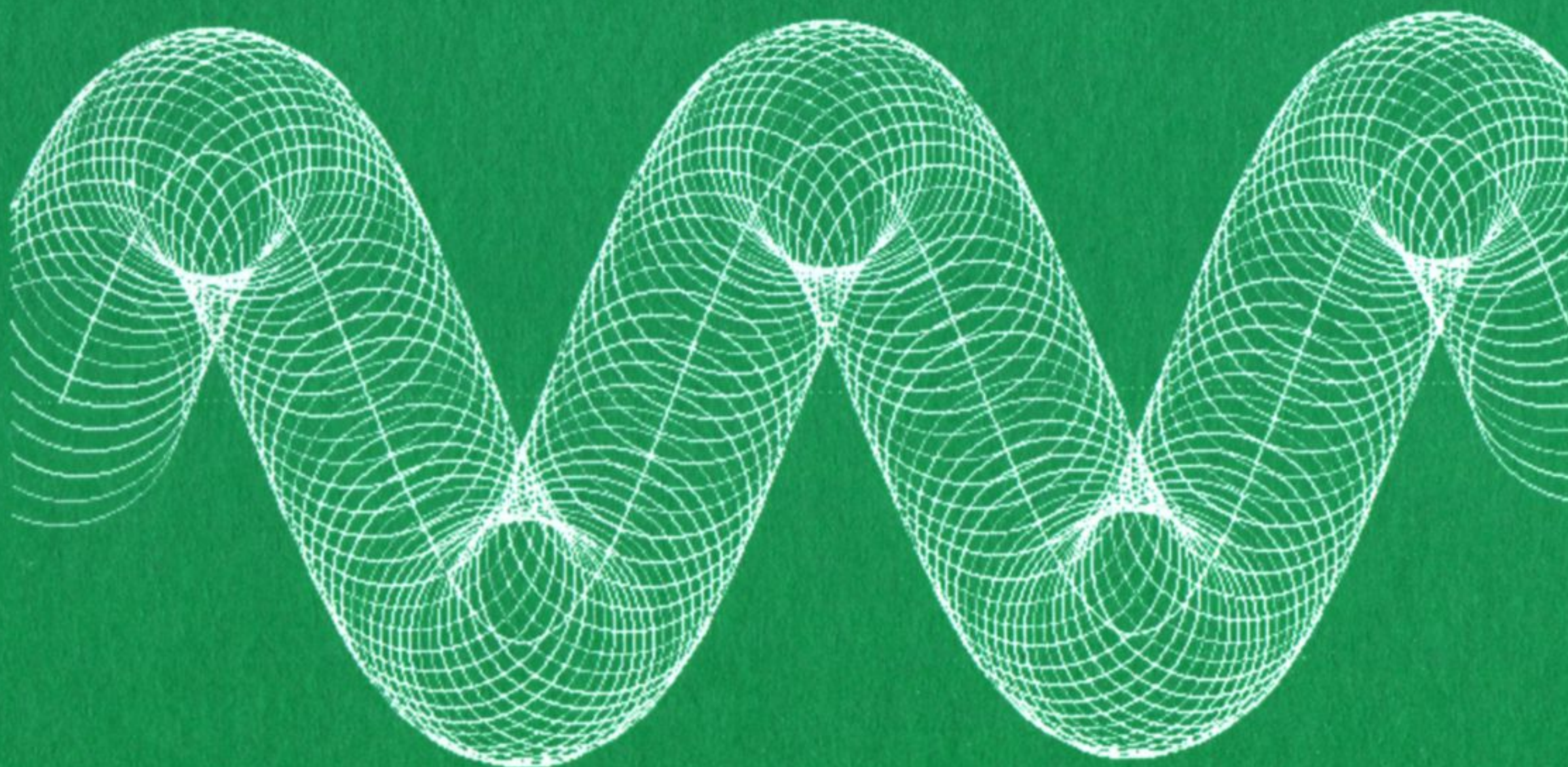
YY=Y1—SIN(X)をYY=Y1—COS(X)に変えることでコサインカーブになります。三角関数には、ほかにTAN（タンゼント）があります。YY=Y1—TAN(X)も試してください。

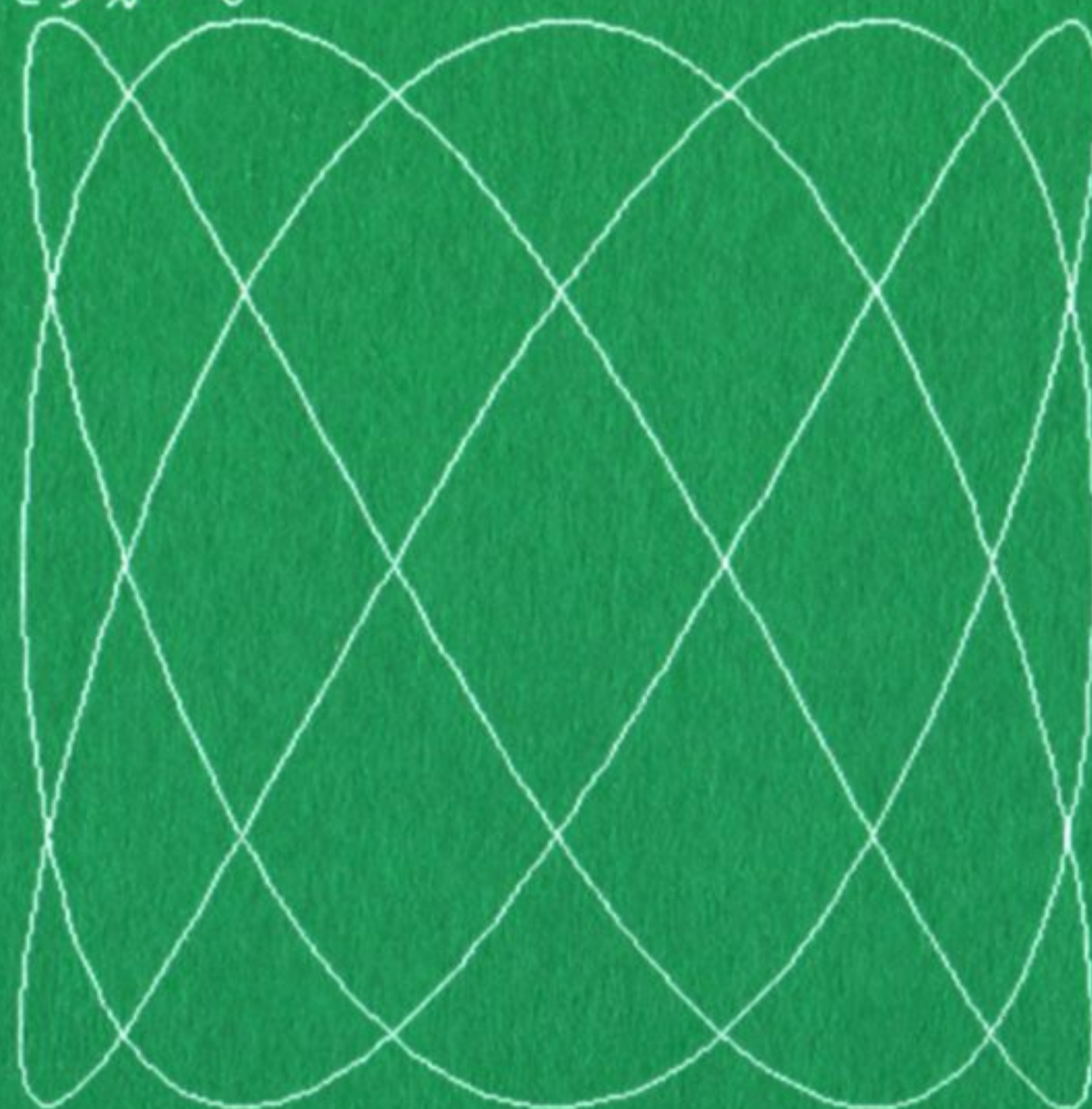
2 サインカーブの応用

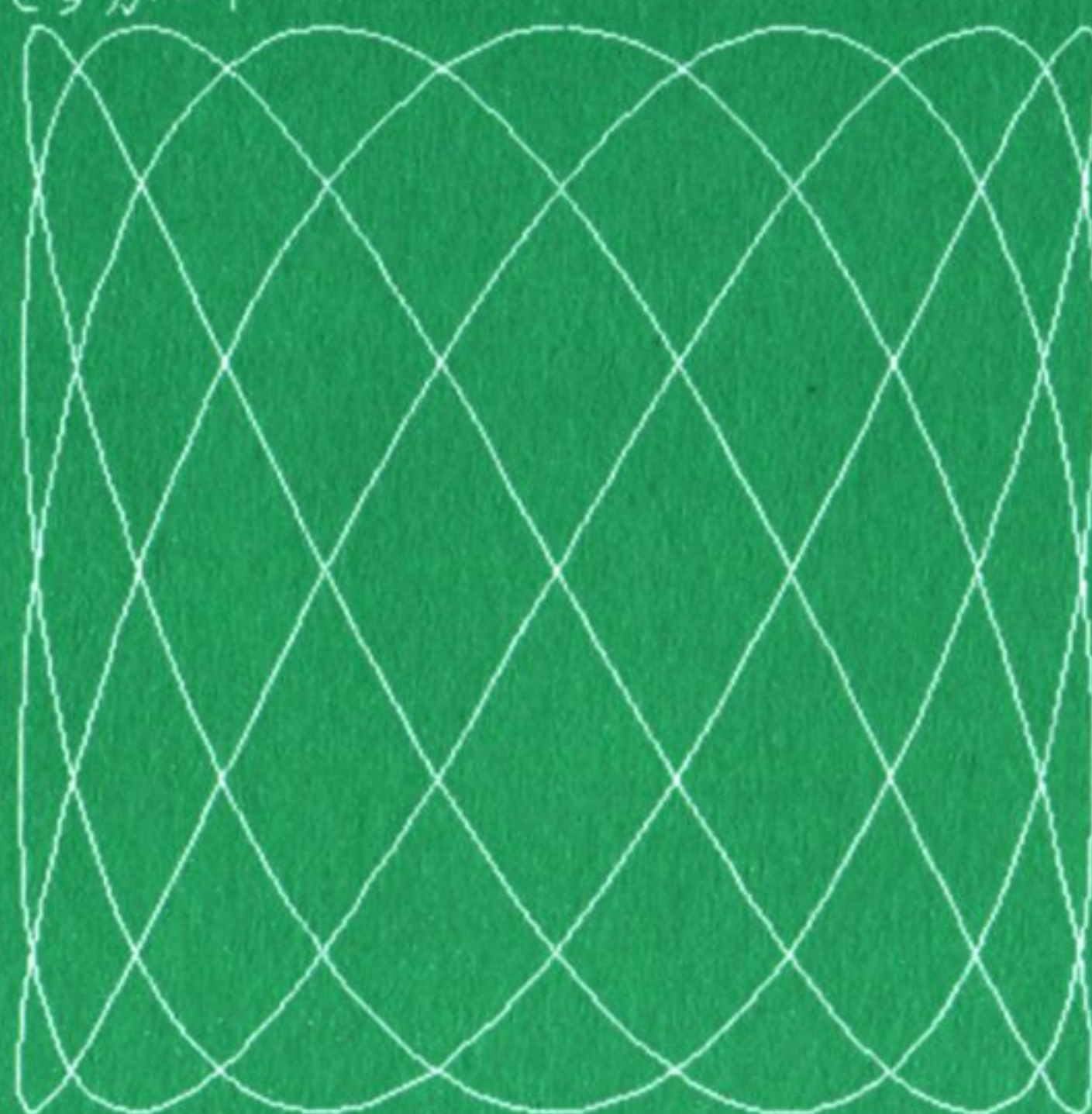
サインカーブに合わせて、円を描いています。また、円の色を変数CLで表していますが、サインカーブのドットの変化と対応させていますので、数字が小数点以下が多いので、いったん10倍して、7で割った余りに1を加える方法をとっています。もし、X*10 MOD 8 だと、0（黒）が入ってくることになります。

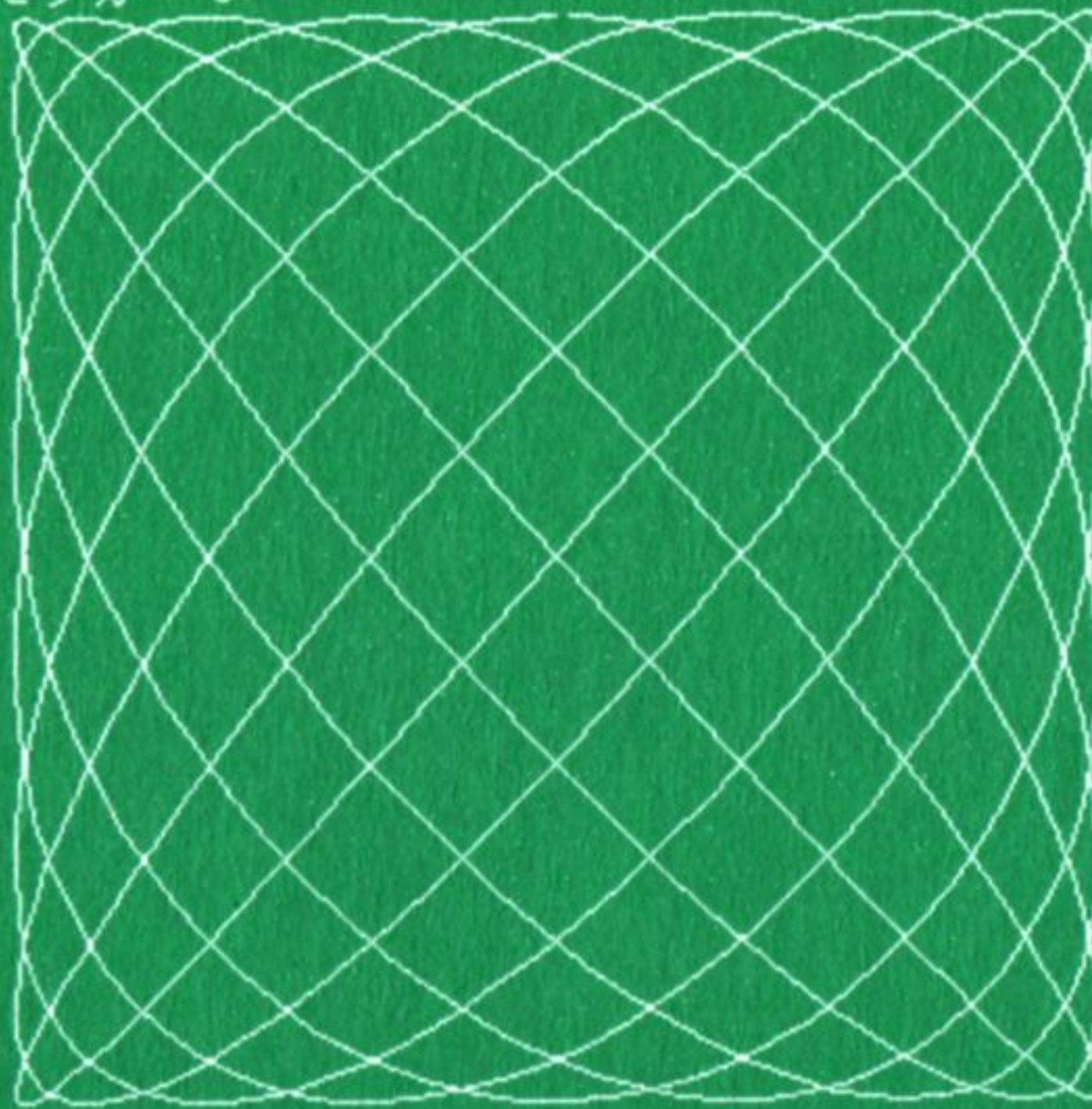
```
'GR-PR308  SIN CURVE
SCREEN 0
CLS
   LINE (0, 200)-(640, 200), 2, , &H8888
       X1 = 20: Y1 = 200
       PI = 3.14159
       X2 = 20: Y2 = 100
       XMI = 0: XMA = 5 * PI
    FOR X = XMI TO XMA STEP .1
            XX = X1 + X2 * X * 2
            yy = Y1 - SIN(X) * Y2
          IF X = XMI THEN
           PSET (XX, yy)
            ELSE
                LINE -(XX, yy)
          END IF
           CL = X * 10 MOD 7 + 1
          CIRCLE (XX, yy), 50, CL
    NEXT
```

<<リサージュ図形>>
上下のくびれはいくつですか＝1
左右のくびれはいくつですか＝2

3 リサージュ図形

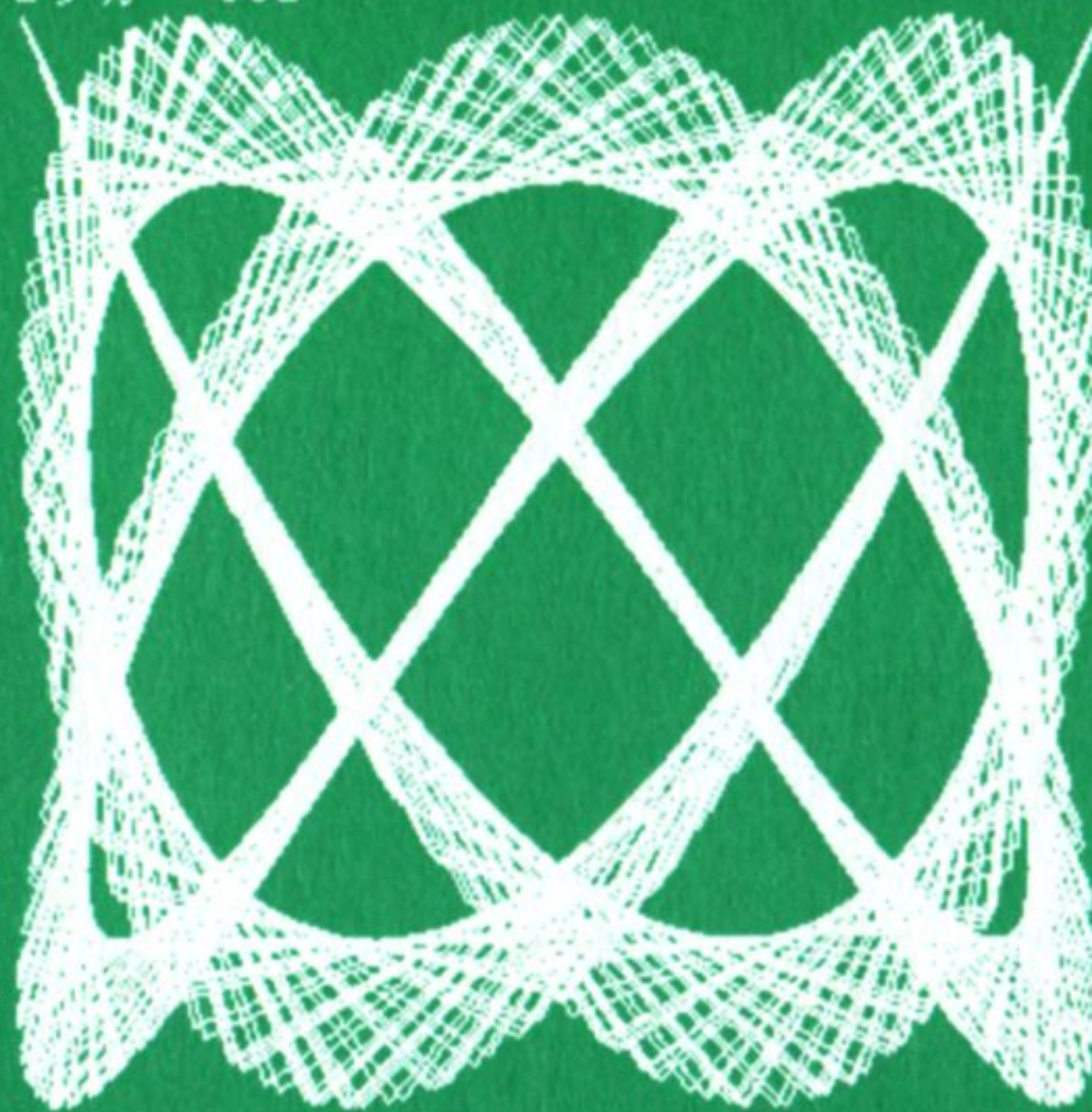
X軸と**Y**軸にサインカーブを加えることで、次のような図形が描かれます。このような図形のことをリサージュ図形とよびます。

何かキーを押してください

<<リサージュ 図形>>
上下のくびれはいくつですか＝3
左右のくびれはいくつですか＝5

何かキーを押してください

<<リサージュ 図形>>
上下のくびれはいくつですか＝4
左右のくびれはいくつですか＝7

何かキーを押してください

<<リサージュ図形>>
上下のくびれはいくつですか=8
左右のくびれはいくつですか=9

何かキーを押してください

```
'PR309.BAS　リサージュ図形
SCREEN 0
CLS
    PRINT "<<リサージュ図形>>"
    INPUT "上下のくびれはいくつですか=", X2
    INPUT "左右のくびれはいくつですか=", Y2

    PI = 3.14159
    X1 = 320: Y1 = 200
    'X2 = 3: Y2 = 5

FOR I = 0 TO 2 * PI STEP .01
    XX = X1 + SIN(I * X2 + PI) * 150
    YY = Y1 - COS(I * Y2) * 150
    IF I = 0 THEN
    PSET (XX, YY)
    ELSE
    LINE -(XX, YY)
    END IF
NEXT
```

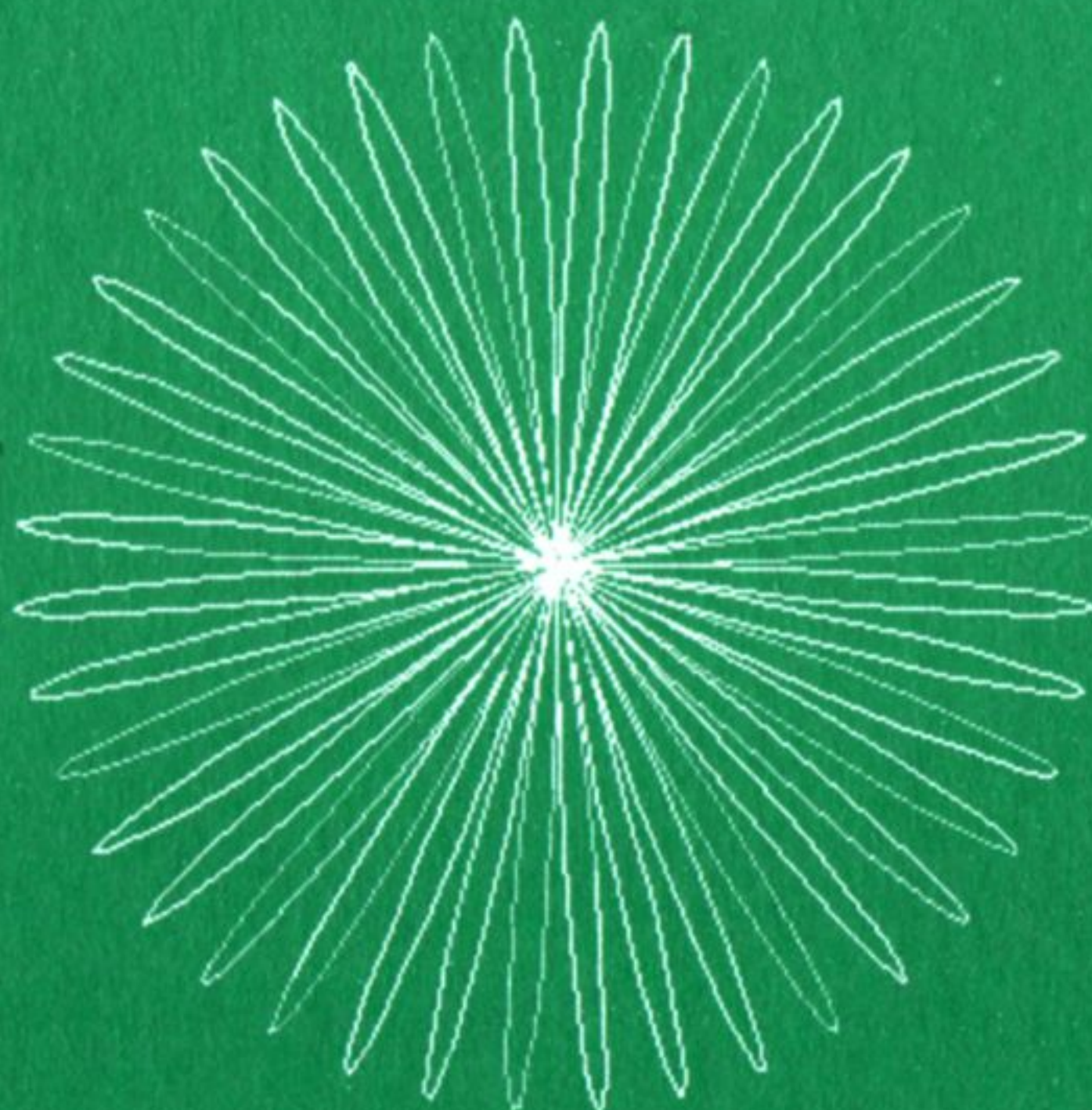
<<リサージュ図形>>
上下のくびれはいくつですか＝100
左右のくびれはいくつですか＝160

何かキーを押してください

4　花びらをつくる

サインとコサインの組み合せです。このプログラムも、三角関数を覚えたてのときの基本的なプログラムになります。

```
'PR310.BAS　花びら
SCREEN 0
CLS
    PRINT "<<花びら図形>>　※偶数は、左右対称の倍の花びらになります"
    INPUT "何枚の花びらにしますか＝", MI
     IF MI MOD 2 = 0 THEN KF = 2 ELSE KF = 1
    PI = 3.14159
    X1 = 320: Y1 = 200

FOR I = 0 TO PI * KF STEP .01

    XX = X1 + SIN(I * MI) * COS(I) * 150
    YY = Y1 - SIN(I * MI) * SIN(I) * 150
```

```
    CL = I * 10 MOD 7 + 1
    IF I = 0 THEN
    PSET (XX, YY), CL
    ELSE
    LINE -(XX, YY), CL
    END IF
NEXT
```

<<花びら図形>>　※偶数は、左右対称の倍の花びらになります
何枚の花びらにしますか＝20

＜試してみよう＞

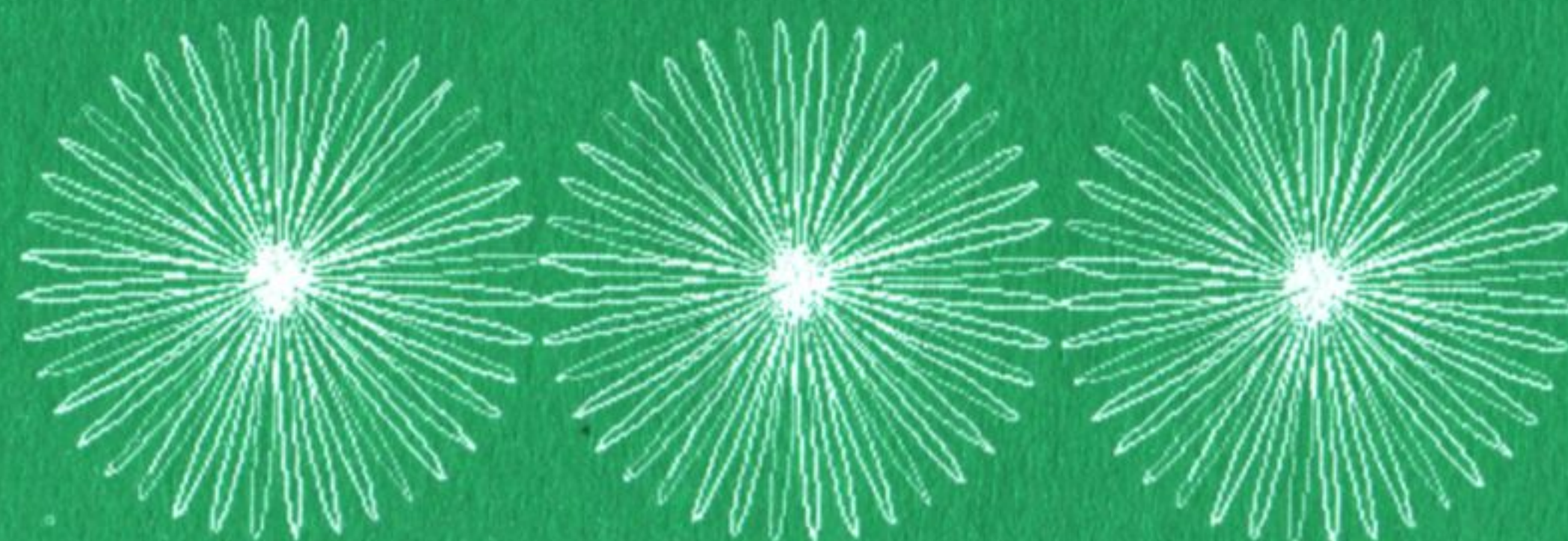
花びらの枚数をn枚。角度Rで0から2πまで変化させたとき、X軸の座標XXと、Y軸の座標YYは、

```
XX=COS(n*R)*SIN(R)
YY=COS(n*R)*COS(R)
```

で、表わすこともできます。例に使ったプログラムとどこがちがって、どこが共通なんでしょうか。結果は同じになるはずですが……。

非常に単純な修飾です。X軸上に3つの花をつくるプログラムの例です。変数X1がFOR...NEXTの中に入っています。表示するX座標に気をつければ、簡単に作ることができます。

```
'PR311.BAS　複数の花びら
SCREEN 0
CLS
    PRINT "<<花びら図形>>　※偶数は、左右対称の倍の花びらになります"
    INPUT "何枚の花びらにしますか=", MI
     IF MI MOD 2 = 0 THEN KF = 2 ELSE KF = 1
    PI = 3.14159
    Y1 = 200: R = 75
 FOR J = 1 TO 3
    FOR I = 0 TO PI * KF STEP .01
            X1 = 150 * J
            XX = X1 + SIN(I * MI) * COS(I) * R
            YY = Y1 - SIN(I * MI) * SIN(I) * R
            CL = I * 10 MOD 7 + 1
        IF I = 0 THEN
            PSET (XX, YY), CL
        ELSE
            LINE -(XX, YY), CL
        END IF
    NEXT
 NEXT
```

```
<<花びら図形>>　※偶数は、左右対称の倍の花びらになります
何枚の花びらにしますか=20
```

第４章　図形の回転

４．１　円の回転

ここでは、図形の回転について整理します。

1　扇形の一覧

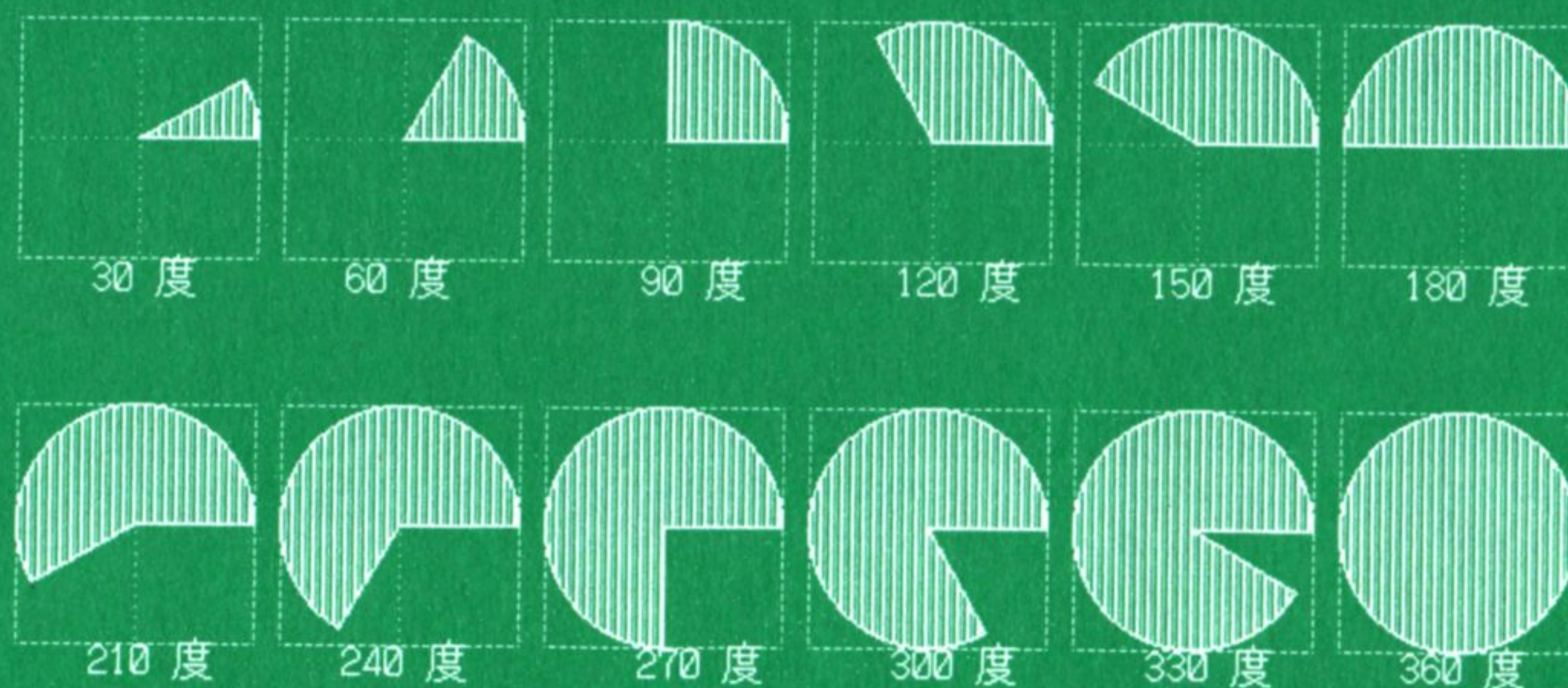
グラフィックスの原点は、つねに点（ドット）単位で考えます。扇形を作るには、始点、終点に負の記号をつける必要があります。そのためには、負の数値が扱える最も小さい数値0.0001を用います。

```
'PR401.BAS 扇形の連続表示
SCREEN 0
CLS
        TI$ = CHR$(&H66) + CHR$(&H55)
        PI = 3.14159
        PRINT "扇形の30度毎の表示"
    FOR XX = 1 TO 6
```

```
    LINE (XX * 100 - 95, 55)-(XX * 100 - 5, 145), 2, B, &HBBBB
    LINE (XX * 100 - 95, 100)-(XX * 100 - 5, 100), 2, , &H8888
    LINE (XX * 100 - 50, 55)-(XX * 100 - 50, 145), 2, , &H8888
    CIRCLE (XX * 100 - 50, 100), 45, , -.00001, -XX * 30 * PI / 180
    PAINT (XX * 100 - 45, 99), TI$
    LOCATE 10, INT((XX * 100 - 60) / 16) * 2: PRINT XX * 30; "度";
    NEXT

    FOR XX = 7 TO 12
LINE ((XX - 6) * 100 - 95, 200)-((XX - 6) * 100 - 5, 290), 2, B, &HBBBB
LINE ((XX - 6) * 100 - 95, 245)-((XX - 6) * 100 - 5, 245), 2, , &H8888
LINE ((XX - 6) * 100 - 50, 200)-((XX - 6) * 100 - 50, 290), 2, , &H8888
CIRCLE ((XX - 6) * 100 - 50, 245), 45, , -.00001, -XX * 30 * PI / 180
    PAINT ((XX - 6) * 100 - 45, 244), TI$
    LOCATE 19, INT(((XX - 6) * 100 - 60) / 16) * 2: PRINT XX * 30; "度";
    NEXT
```

扇形の30度毎の表示

30 度　60 度　90 度　120 度　150 度　180 度

210 度　240 度　270 度　300 度　330 度　360 度

2　カラードーナツ

CIRCLE文と円の方程式を使って、半円をずらしながらSTEPに合わせてカラーコードを作り出します。ドーナツの大きさは変数Rの値を変えてやります。

```
'PR402.BAS ドーナツ
SCREEN 0
CLS
        PI = 3.14159: R = 150
FOR X = 0 TO 360 STEP 1
    XX = R * COS(X * PI / 180) + 320
    YY = 200 - R / 2 * SIN(X * PI / 180)
    CL = X * 10 MOD 8
    S1 = 180 + X
        IF S1 > 360 THEN S1 = S1 - 360
    E1 = X
        IF E1 > 360 THEN E1 = E1 - 360
    S1 = S1 * PI / 180: E1 = E1 * PI / 180

    CIRCLE (XX, YY), 50, CL, S1, E1, .5
NEXT
        LOCATE 13, 34
        PRINT "ドーナツ図形"
 END
```

ドーナツ図形

<試してみよう>

ドーナツプログラムのSTEPの値XXを変化させてみてください。

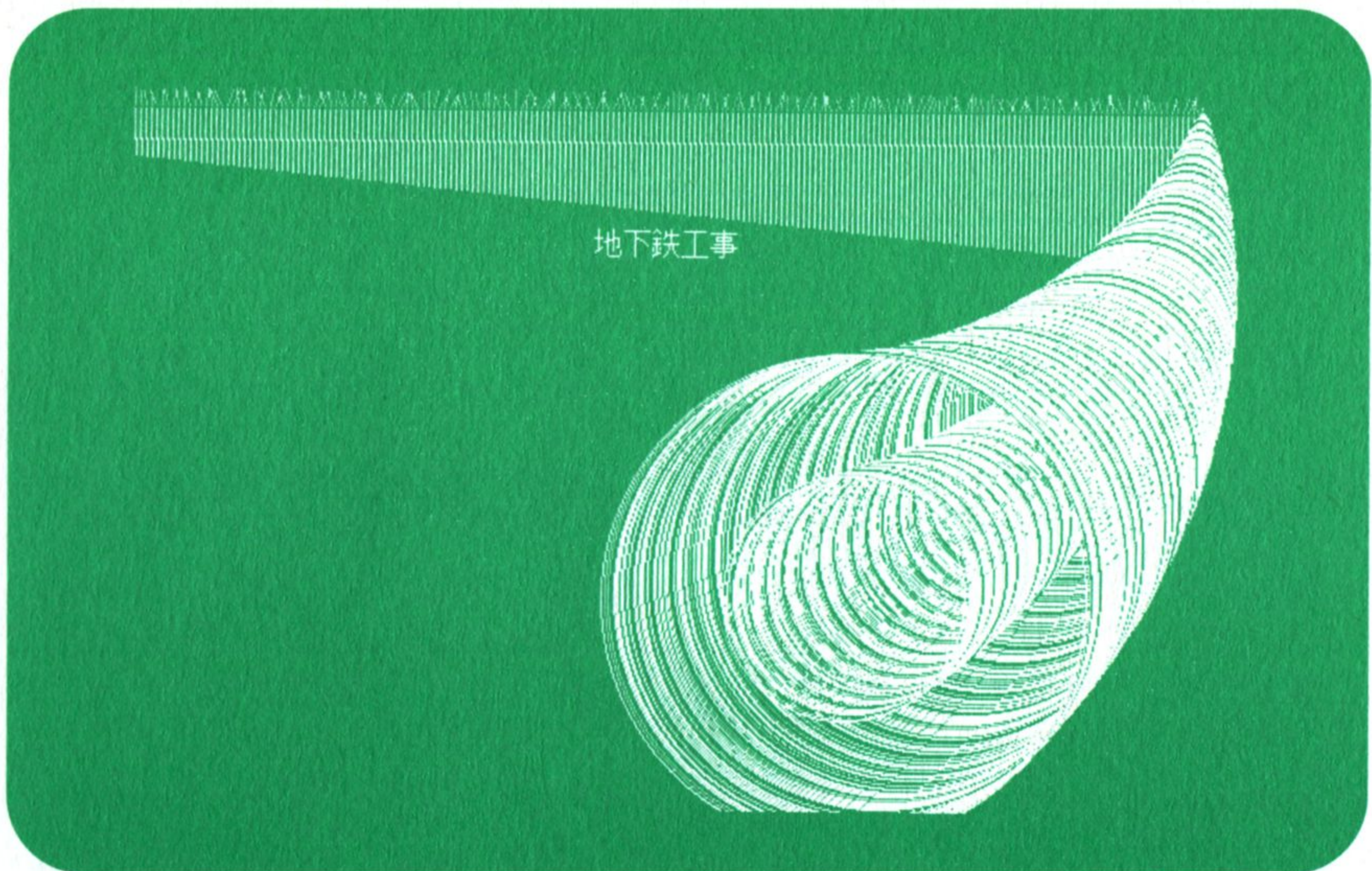
```
FOR X=0 TO 360 STEP XX
```

ドーナツの色も変化します。どうしてでしょう？

3 地下鉄工事

東京・御徒町(おかちまち)で起こった新幹線工事の道路陥没は、車を呑み込みけが人を出しましたが、この地下鉄工事は安全です。地上の草は乱数を使っています。

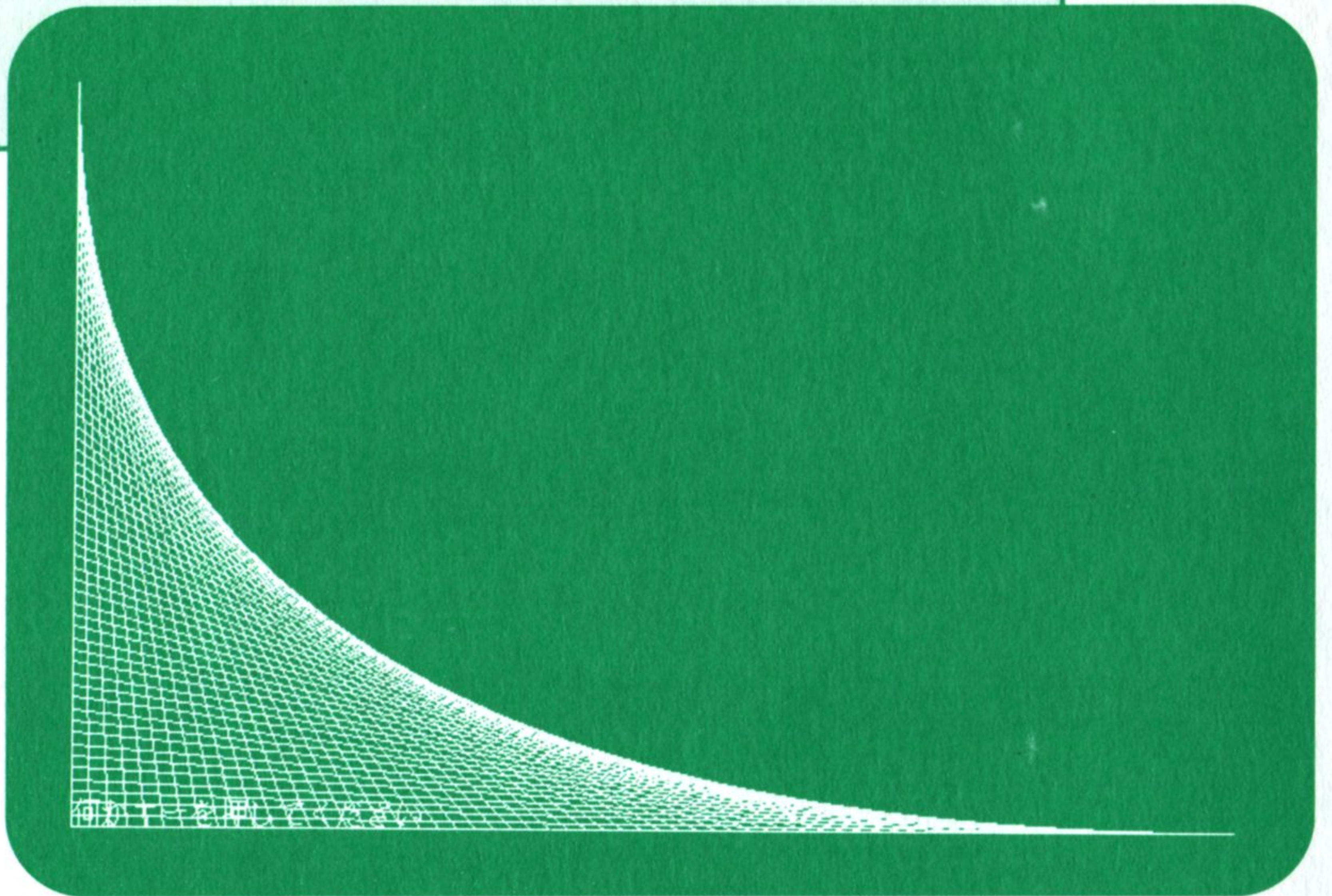
```
'PR403.BAS 地下鉄工事
SCREEN 0
CLS
    RANDOMIZE VAL(RIGHT$(TIME$, 2))
FOR T = 10 TO 600 STEP 3
      LINE (T, 15)-(T, 32), 4
      LINE (T + 1, 32)-(T + 1, 40 + T / 10), 6
      KT = INT(RND(1) * 5): GKF = INT(RND(1) * 2) + 1
      KU = INT(RND(1) * 3): KSF = INT(RND(1) * 2) + 1
      IF GKF = 1 THEN KT = -KT
      IF KSF = 1 THEN KU = -KU
      LINE (T, 13)-(T + KT, 5 + KU), 2
NEXT
        LOCATE 6, 34: PRINT "地下鉄工事"

        PI = 3.14159
FOR X = 2 TO 66 STEP .1
        XX = RJ * COS(X * PI / 180) + 10
        YY = 600 - RJ * SIN(X * PI / 180)
            CL = X * 10 MOD 8
        CIRCLE (YY, XX), 2 * R, CL
        CIRCLE (YY, XX), R, CL
            RJ = RJ + 1: R = R + .2
NEXT
END
```

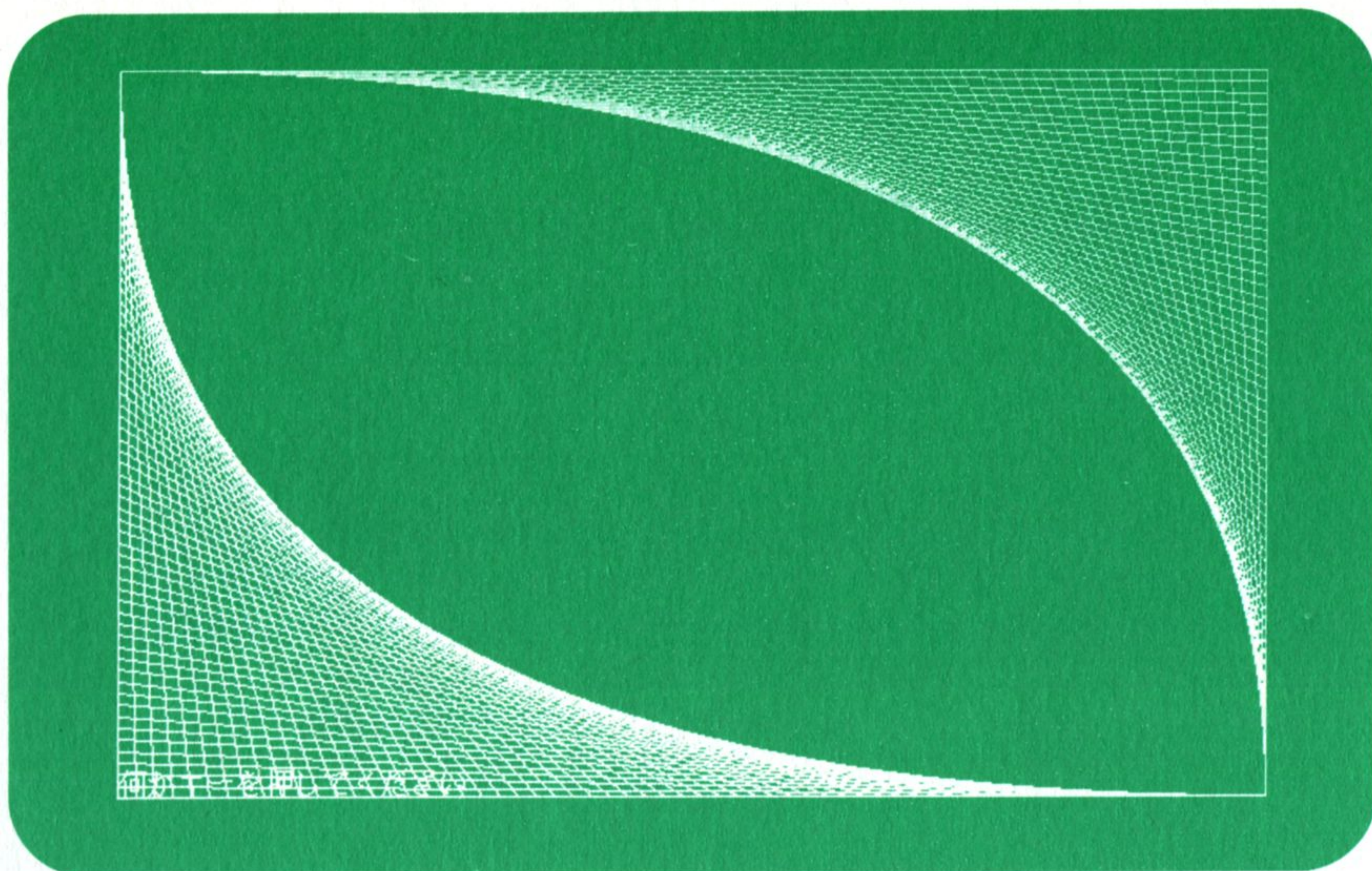
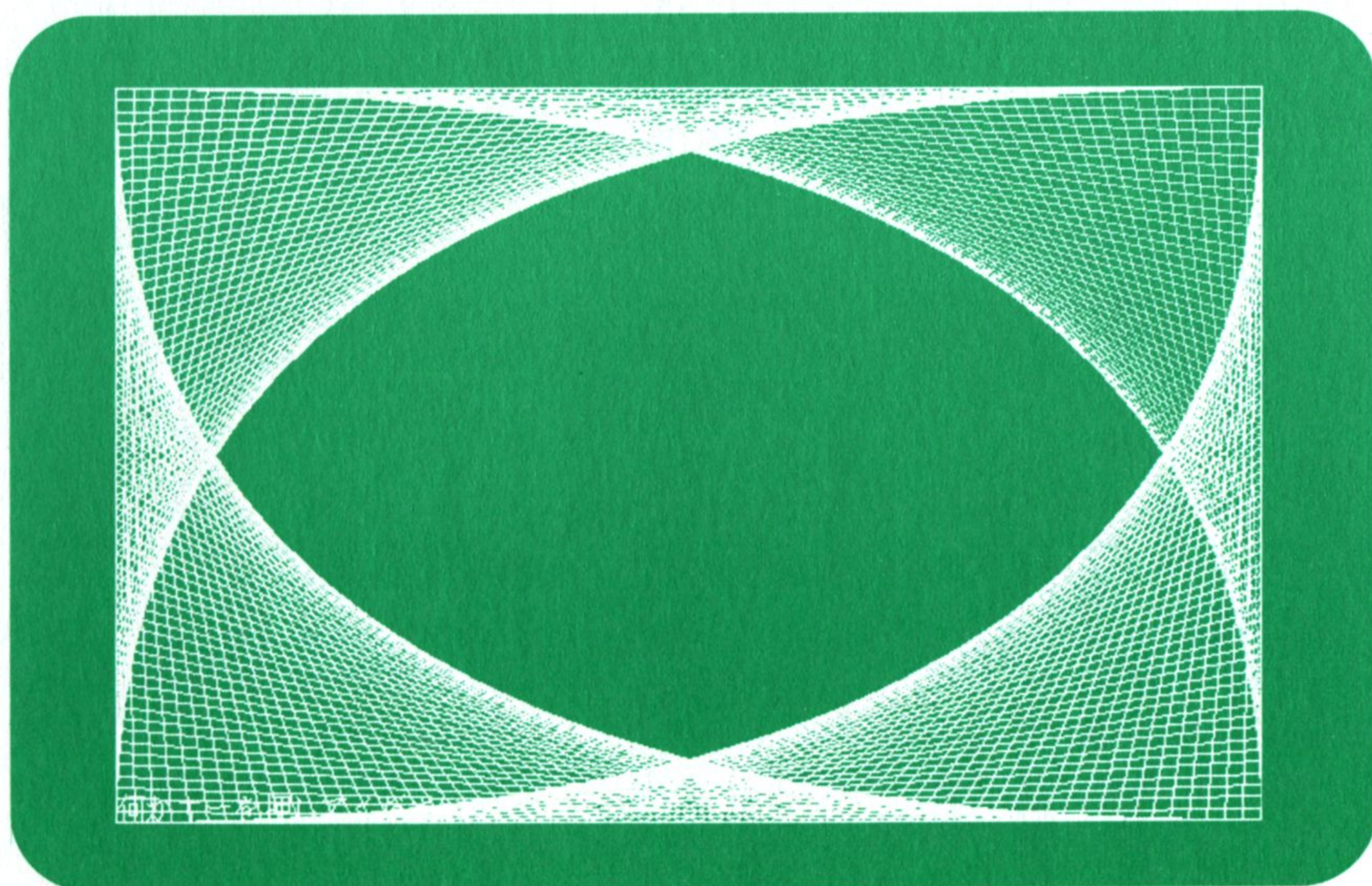
地下鉄工事

4.2 直線を使った曲線

直線だけを使って、グラフィックスの作成を行ないます。究極の多角形が円であるのと同じように、直線を集合させるだけで曲線を作ることができます。

1　2個の始点をもつ直線の集合

X軸0点を、最初の始点、Y軸399を第2の始点とします。COLORコードは、5(紫)になっています。このプログラム例で、X軸、Y軸の位置関係をしっかりつかむようにします。

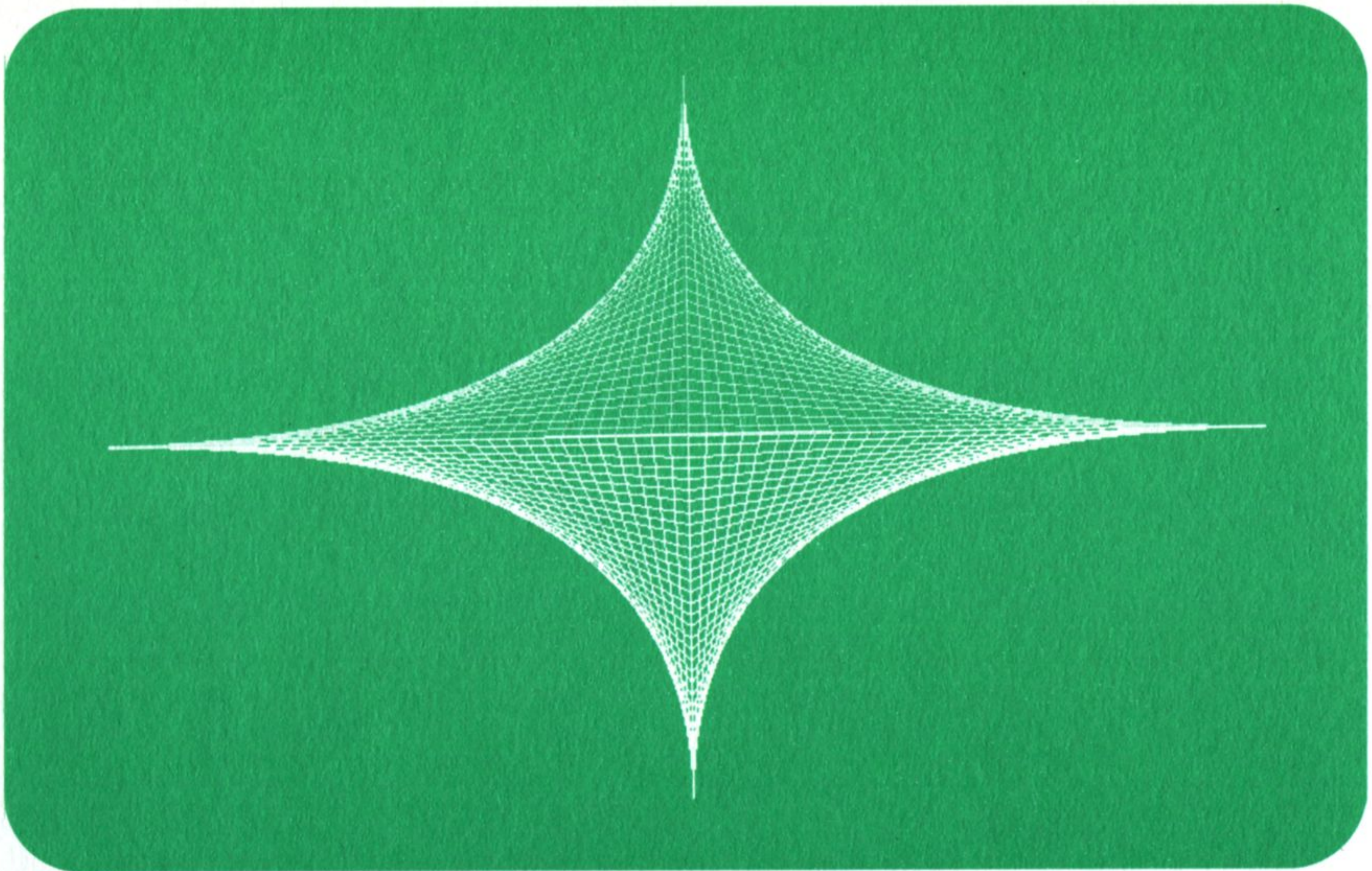
```
'PR404.BAS 直線
SCREEN 0
CLS
    FOR X = 0 TO 630 STEP 10
        LINE (0, X / (630 / 399))-(X, 399), 5
    NEXT
END
```

単純に直線の応用例です。画面右下に紫と画面左上に緑の直線を使った曲線を表示します。LINE文がひとつ増えています。さらに、LINE文を増やして、4つの直線を使った曲線を作っています。各直線のXの値の変化とYの値の変化を計算してください。

```
'PR405.BAS 対称直線
SCREEN 0
CLS
    FOR X = 0 TO 630 STEP 10
        LINE (0, X / (630 / 399))-(X, 399), 5
        LINE (630, 399 - (X / (630 / 399)))-(630 - X, 0), 2
    NEXT
END
```

▲PR405.BASの実行画面

▲PR406.BASの実行画面

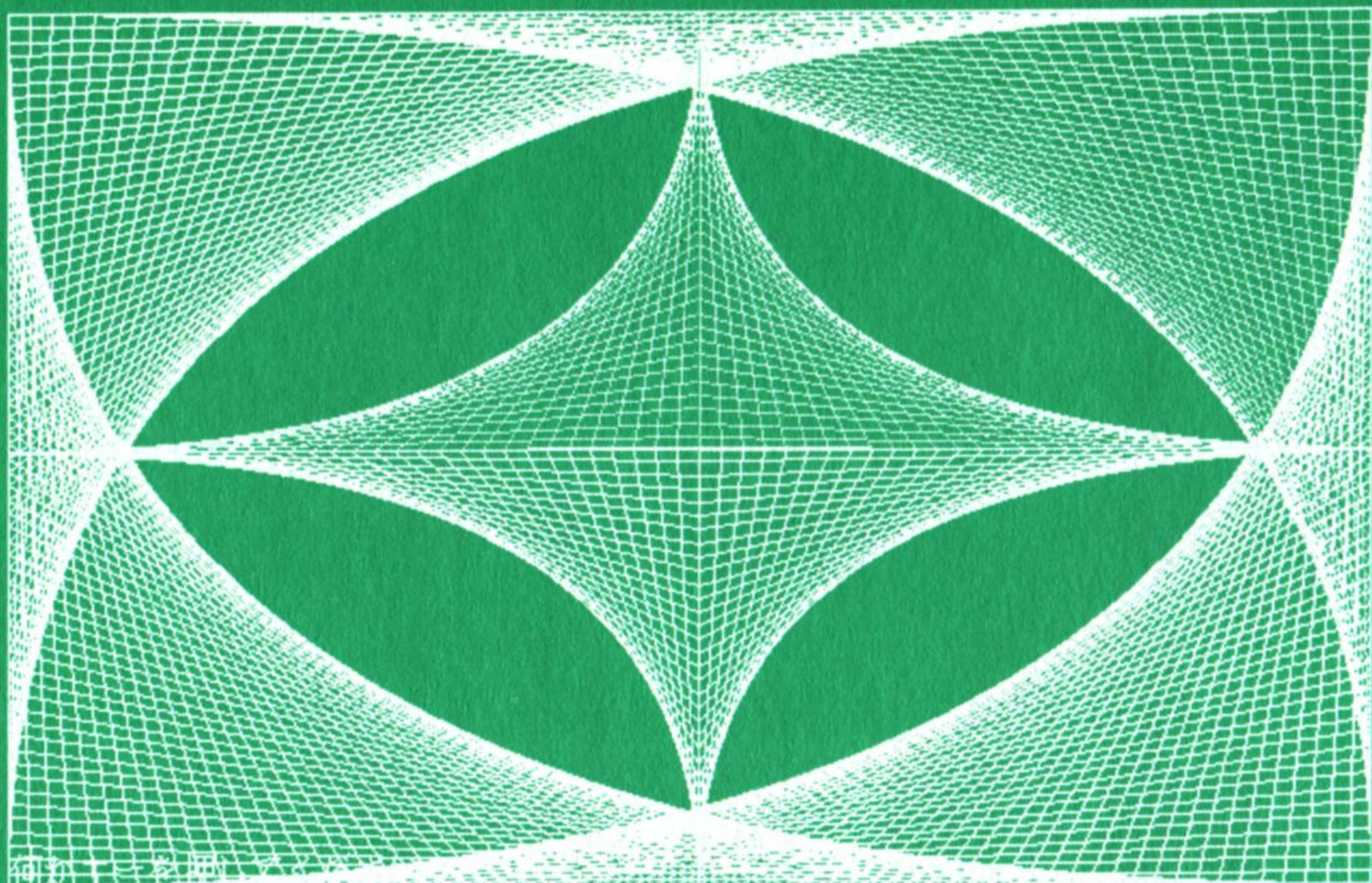
```
'PR406.BAS 対称直線
SCREEN 0
CLS
    FOR X = 0 TO 630 STEP 10
        LINE (0, X / (630 / 399))-(X, 399), 5
        LINE (630, 399 - (X / (630 / 399)))-(630 - X, 0), 2
        LINE (630, X / (630 / 399))-(630 - X, 399), 7
        LINE (0, 399 - (X / (630 / 399)))-(X, 0), 4
    NEXT

END
```

2 直線の応用による曲線

直線を使った曲線は、始点を画面4端に置いていました。その始点を（320,200）の1点に集めたのが、このプログラムです。カラーコード順に画面に作成されます。

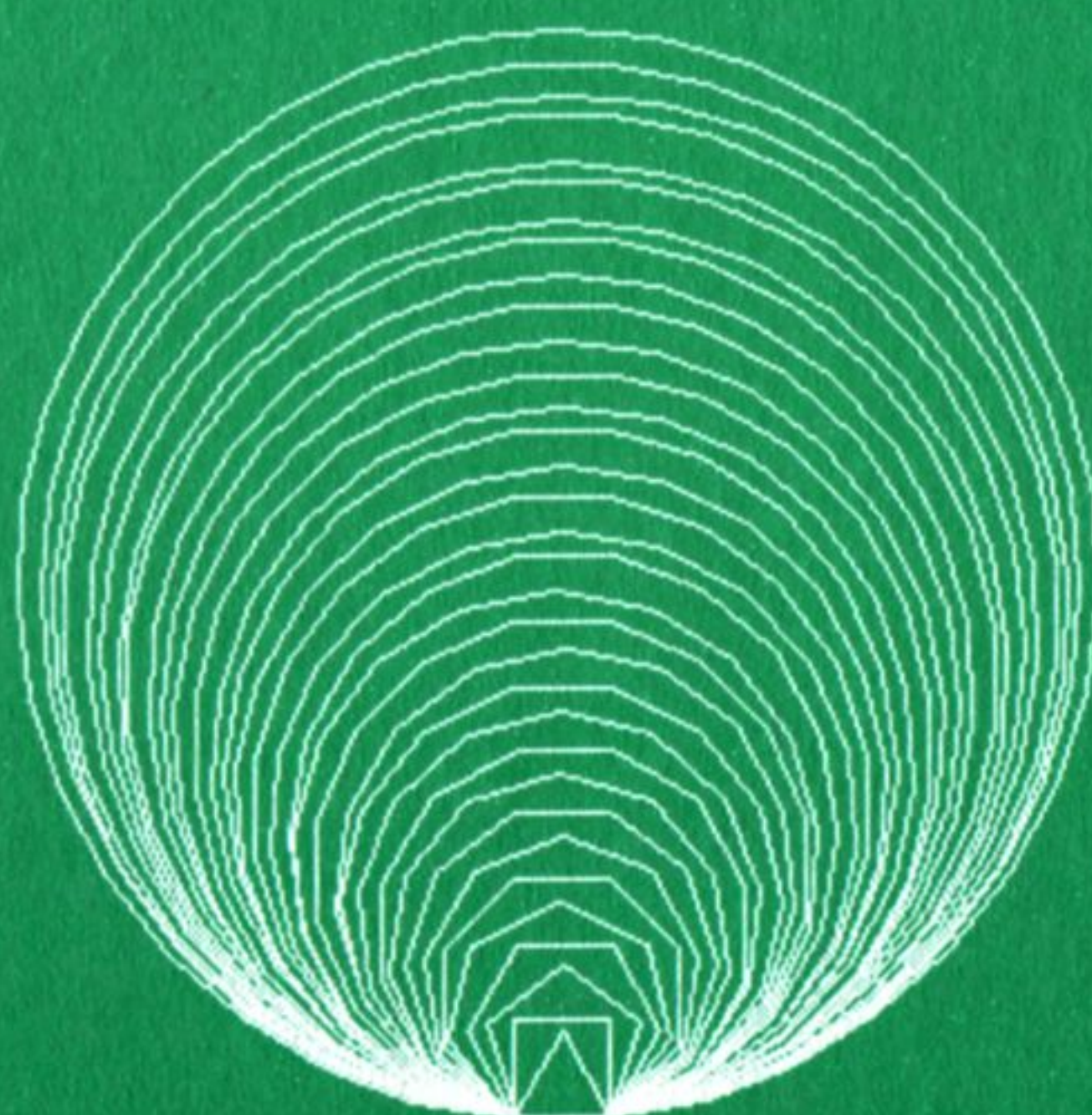
```
'PR407.BAS 対称直線
SCREEN 0
CLS
    FOR X = 0 TO 320 STEP 10
        LINE (320, X / (320 / 200))-(320 - X, 200), 1
        LINE (320, X / (320 / 200))-(X + 320, 200), 2
        LINE (320, 399 - (X / (320 / 200)))-(320 - X, 201), 3
        LINE (320, 399 - (X / (320 / 200)))-(X + 320, 201), 4

    NEXT
END
```

ただ、ひたすらLINE文を増やして作ったプログラムです。すべて、XとYの始点をずらしています。直線を使った曲線を画面の4端と中央に始点を置いて組み合わせただけです。

```
'PR408.BAS 対称直線
SCREEN 0
CLS
    FOR X = 0 TO 630 STEP 10
        LINE (0, X / (630 / 399))-(X, 399), 5
        LINE (630, 399 - (X / (630 / 399)))-(630 - X, 0), 2

        LINE (630, X / (630 / 399))-(630 - X, 399), 7
        LINE (0, 399 - (X / (630 / 399)))-(X, 0), 4
    NEXT


    FOR X = 0 TO 320 STEP 10
        LINE (320, X / (320 / 200))-(320 - X, 200), 1
```

```
        LINE (320, X / (320 / 200))-(X + 320, 200), 2
        LINE (320, 399 - (X / (320 / 200)))-(320 - X, 201), 3
        LINE (320, 399 - (X / (320 / 200)))-(X + 320, 201), 4

    NEXT
END
```

3　多角形を作る

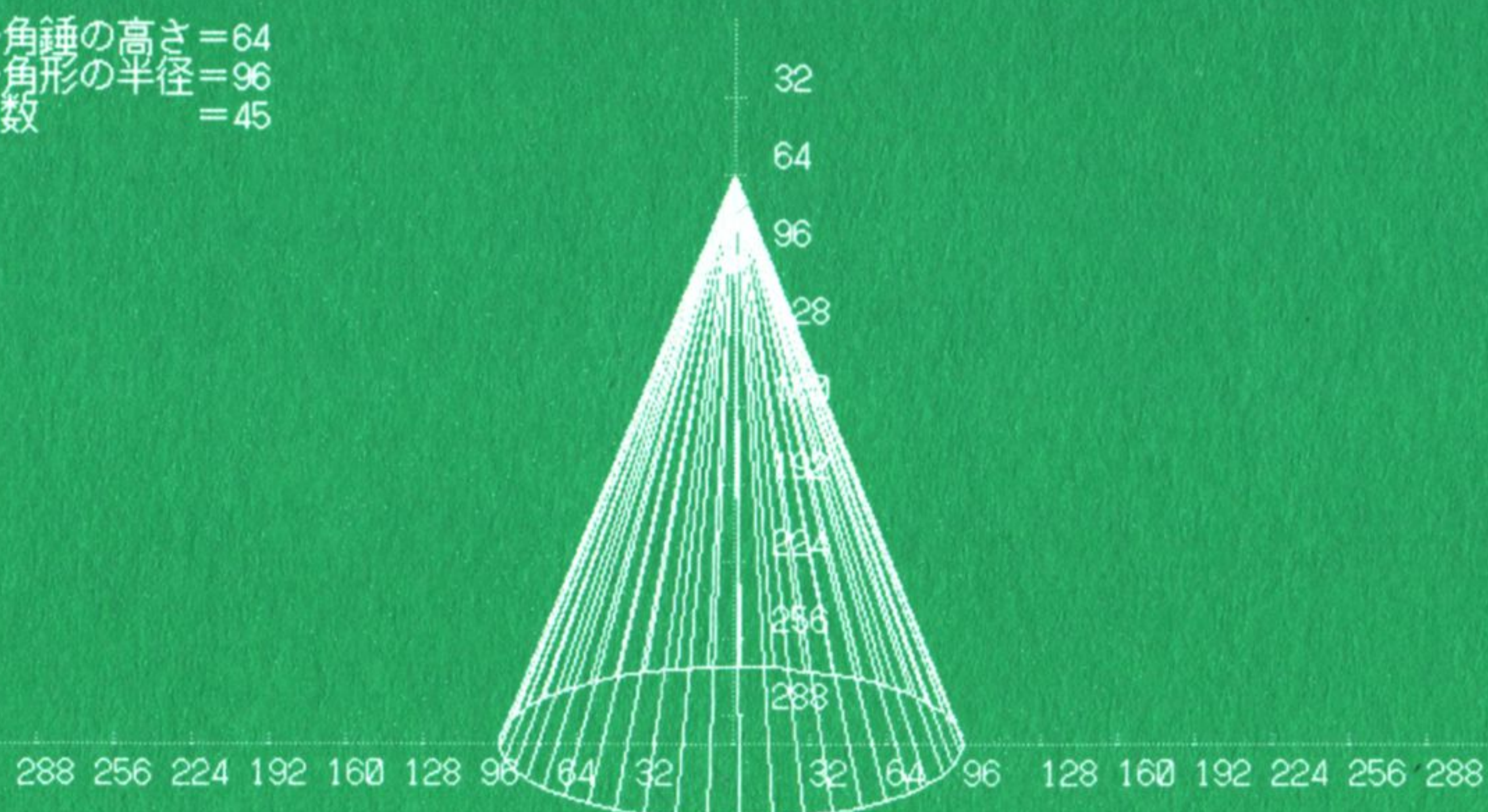
このプログラムは、多角形を作るサンプルプログラムです。Nの値が、最終的な多角形の数です。N=7だとかN=15など、実際の机の上では描きにくい多角形を作ります。パソコンの画面の解像度の関係で、N=35では円に見えます。

```
'PR409.BAS 線の増殖
SCREEN 0
CLS
    N = 35: X1 = 320: Y1 = 350: R = 50: PI = 3.14159
    FOR J = 2 TO N
    PSET (X1, Y1), 4
    RJ = 2 * PI / J
```

```
    FOR I = 0 TO J - 1

    XX = COS(I * RJ) * R
    YY = SIN(I * RJ) * R
    LINE -STEP(XX / 2, -YY / 2)

  NEXT
    NEXT
    END
```

多角形の応用です。画面上にドット単位の目盛りを表示して、多角錘を作るのに必要なデータを画面から入力します。

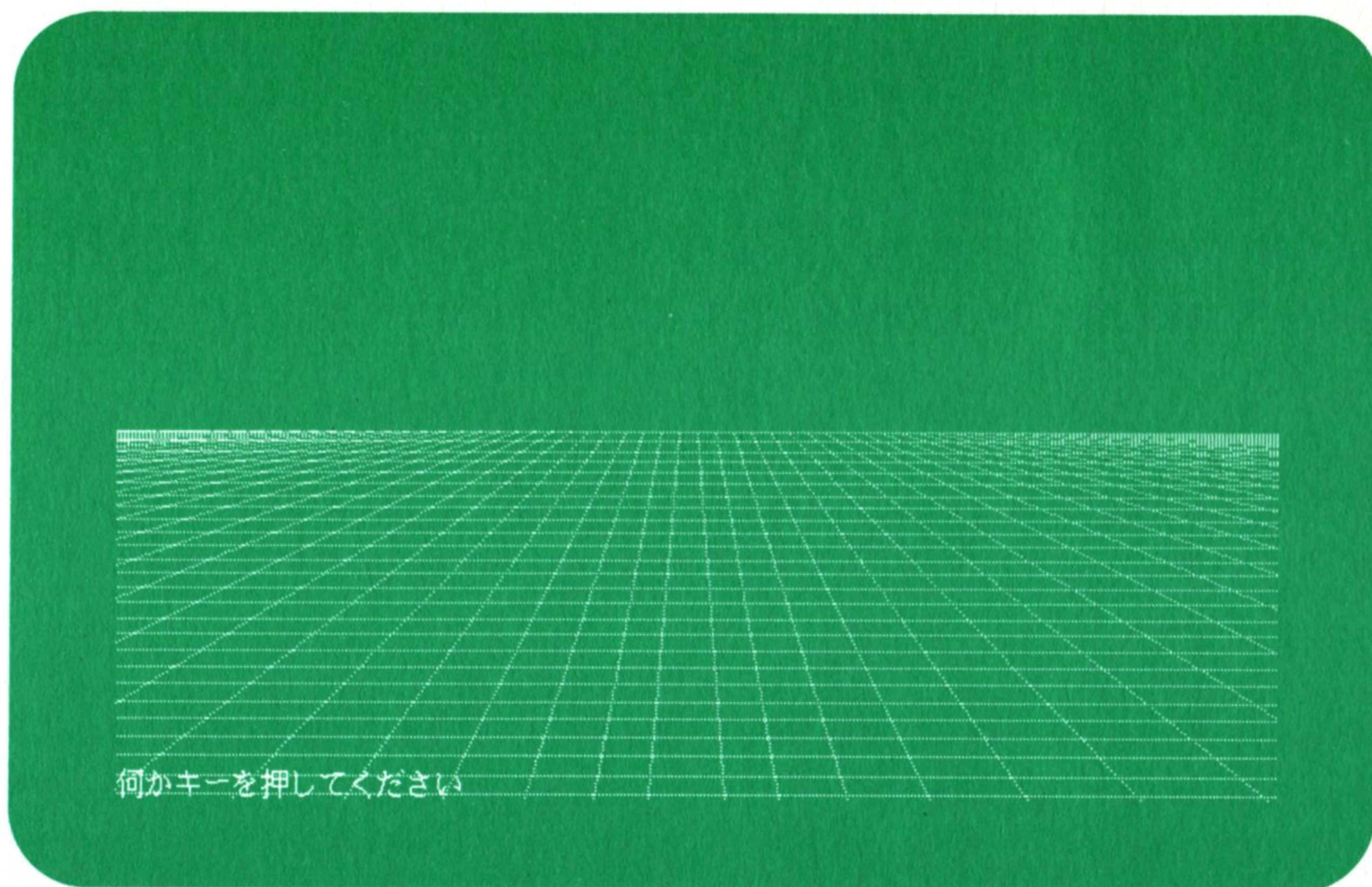
```
'PR410.BAS 多角錘
SCREEN 0
CLS
        LINE (320, 0)-(320, 300), 2, , &HAAAA
        LINE (0, 300)-(640, 300), 2, , &HAAAA
    FOR YM = 1 TO 9
        LINE (316, YM * 32)-(324, YM * 32), 2, , &HAAAA
        LINE (320 + YM * 32, 296)-(320 + YM * 32, 300), 2, , &HAAAA
        LINE (320 - YM * 32, 296)-(320 - YM * 32, 300), 2, , &HAAAA
        LOCATE YM * 2, 42: PRINT YM * 32
        LOCATE 20, 39 - YM * 4: PRINT YM * 32
        LOCATE 20, 40 + YM * 4: PRINT YM * 32
```

```
    NEXT
        LOCATE 1, 1
            INPUT "多角錘の高さ=", Y0
            INPUT "多角形の半径=", R
            INPUT "角数        =", KAK
        X = 320: Y = 300: X0 = 320
        XR = 1: YR = 3: PI = 3.14159: K = 360 / KAK
        DIM X1(KAK), Y1(KAK)
FOR I = 0 TO 360 STEP K
    FOR J = 1 TO KAK
        C = PI / 180 * (I + K * (J - 1))
        X1(J) = X + R / XR * COS(C)
        Y1(J) = Y - R / YR * SIN(C)
    NEXT

    FOR J = 1 TO 2
        H = J + 1
        LINE (X0, Y0)-(X1(J), Y1(J)), 5
            PSET (X1(J), Y1(J)), 5
        LINE -(X1(H), Y1(H)), 7
    NEXT
NEXT

END
```

多角錘の高さ=64
多角形の半径=96
角数　　　　=45

4　階乗式を使った直線

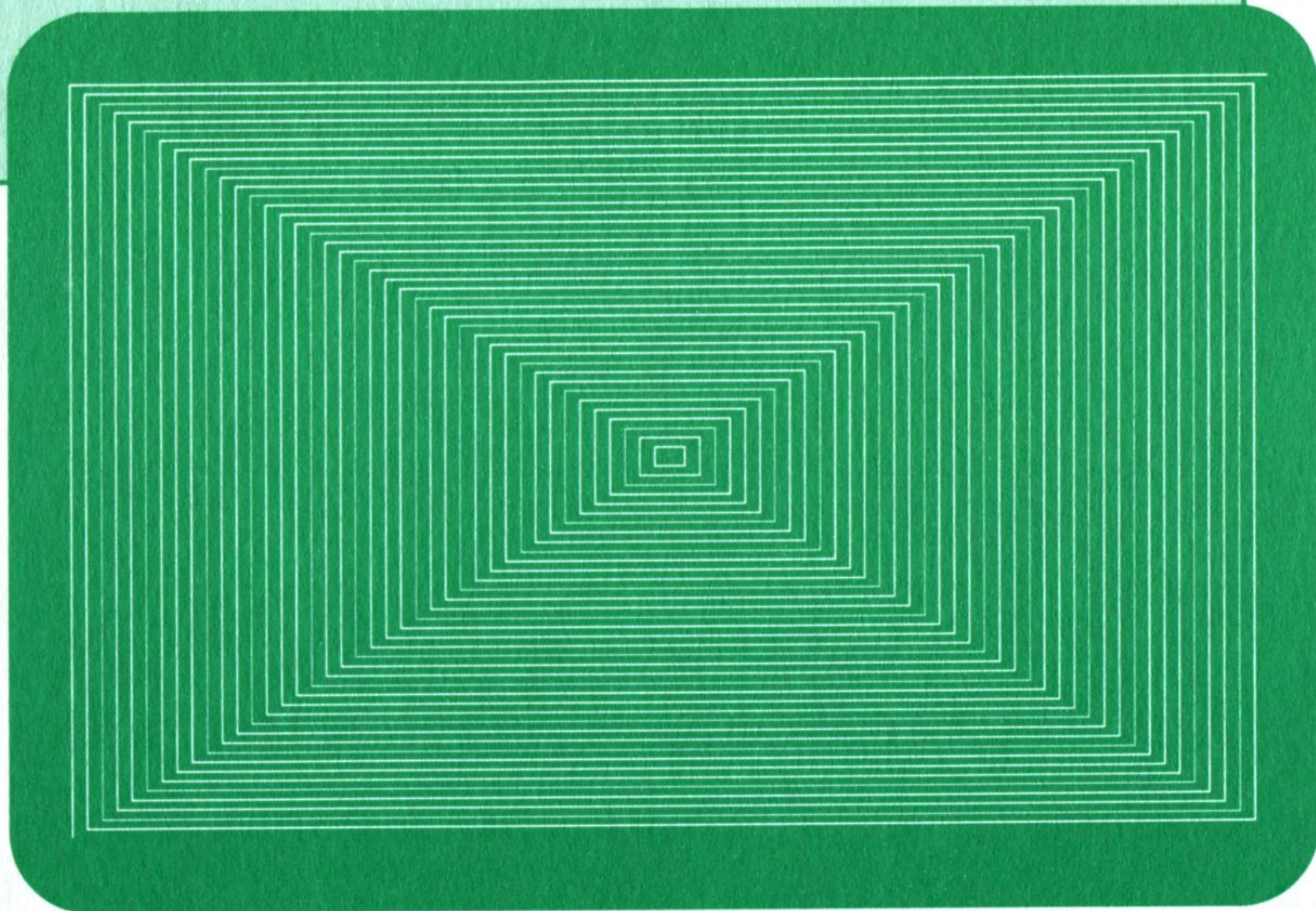
3つのLINE文を組み合わせます。どのプログラムが、画面上にどのように表示されているかは、LINE文のあとのカラーコード3を、1（青）、2（緑）、3（水色）と変えてやるのもひとつの方法です。

```
'PR411.BAS 直線
SCREEN 0
CLS
        N = 0
    FOR Y = 200 TO 400 STEP 5
        LINE (0, Y)-(640, Y), 3, , &HAAAA
            N = N + 1
            Y = Y + N * 2 / 3 ^ 2
    NEXT
        N = 2
    FOR X = 310 TO 0 STEP -10
            N = N + 2
        LINE (X, 200)-(X - N ^ 2, 400), 3, , &HAAAA
    NEXT
        N = 2
    FOR X = 320 TO 640 STEP 10
            N = N + 2
        LINE (X, 200)-(X + N ^ 2, 400), 3, , &HAAAA
    NEXT
END
```

何かキーを押してください

5 息をするカラー四角形

四角形が色を変えながら画面いっぱいに膨らみやがて、中央へしぼんでいくようすを10回繰り返します。このプログラムの中に、「'LINE(...」とREMをつけたLINE文があります。このREMを取るとどうなるのでしょう。たったひとつのREMで実行結果のイメージが変わります。これが、プログラムの楽しさでもあります。

```
'PR412.BAS 矩形
SCREEN 0
CLS
 FOR I = 0 TO 10
    K = 320 / 200
    FOR Y = 0 TO 200 STEP 5
    CL = Y MOD 7 + 1
        LINE (320 - Y * K, 200 - Y)-(320 + Y * K, 200 + Y), CL, B
      '  LINE (320 - Y * K, 200 - Y)-(320 + Y * K, 200 + Y), 0, B
    NEXT

    FOR Y = 200 TO 0 STEP -5
```

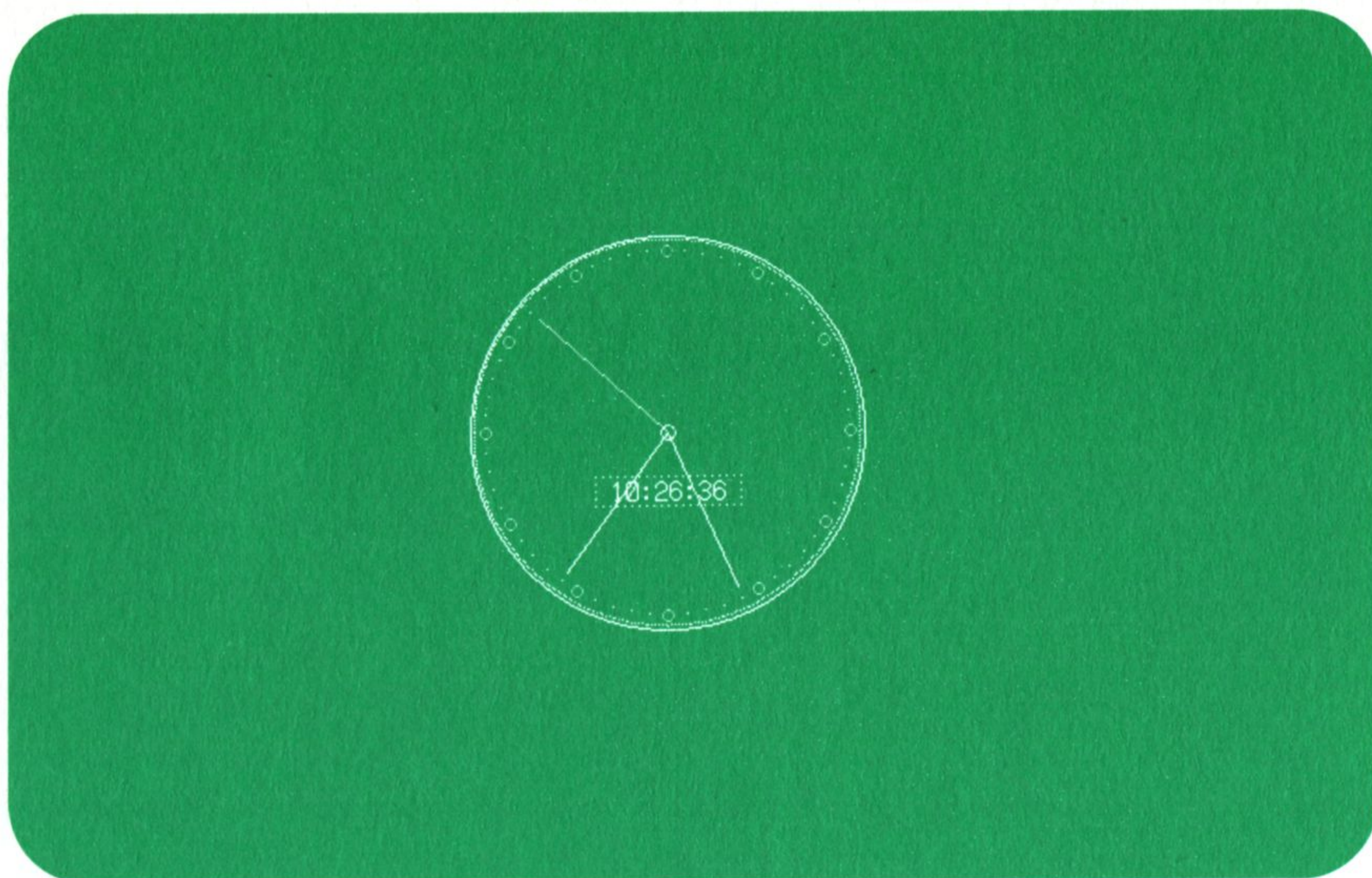
```
    CL = Y MOD 7 + 1
       ' LINE (320 - Y * K, 200 - Y)-(320 + Y * K, 200 + Y), CL, B
        LINE (320 - Y * K, 200 - Y)-(320 + Y * K, 200 + Y), 0, B
    NEXT
  NEXT
END
```

6　中三針時計

直接コマンドでPRINT TIME$と実行すれば、画面上に、たとえば、13:42:51と表示されます。これは、パソコンの内部時計が午後1時42分51秒を示しています。

この内部時間の変化を三角関数と組み合わせると中三針のアナログ時計となります。針の長さや太さ、時計の文字盤など、必要最小限度の表示しかしていません。針の表示の方法は、GET、PUTコマンドを使うと、もっと美しいかわいらしい時計を作ることができます。

```
'PR413.BAS 時計
SCREEN 0
CLS
    DIM XX(360), YY(360): PI = 3.14159
    CIRCLE (320, 200), 108, 7
    R = 106: S = -1: GOSUB HYOJI:
```

```
    R = 100: S = -6: FL = 2: GOSUB HYOJI:
    R = 94: S = -6: FL = 1: GOSUB HYOJI:
        DO
            GOSUB KEISAN:
        LOOP

HYOJI:
        J = -1
    FOR I = 450 TO 91 STEP S
        J = J + 1
            XX(J) = INT(R * COS(I * PI / 180)) + 320
            YY(J) = INT(200 - R * SIN(I * PI / 180))
        IF FL = 2 AND J MOD 5 = 0 THEN
            CIRCLE (XX(J), YY(J)), 3, 2
        ELSEIF FL <> 1 THEN
            PSET (XX(J), YY(J))
        END IF
    NEXT
 RETURN

KEISAN:
                M = VAL(RIGHT$(TIME$, 2))
            S = VAL(MID$(TIME$, 4, 2))
        T = VAL(LEFT$(TIME$, 2))
    IF T >= 13 THEN T = T - 12
        TT = T * 5 + S / 12
                LINE (320, 200)-(XX(M), YY(M)), 7
            LINE (320, 200)-(XX(S), YY(S)), 6


        LINE (320, 200)-(XX(TT), YY(TT)), 4
                LOCATE 15, 37: PRINT TIME$
            LINE (35 * 8, 14 * 16)-(45 * 8, 15 * 16), 7, B, &H8888
                LINE (320, 200)-(XX(M), YY(M)), 0
            LINE (320, 200)-(XX(S), YY(S)), 0
        LINE (320, 200)-(XX(TT), YY(TT)), 0
            CIRCLE (320, 200), 4, 4
RETURN
END
```

10:26:36

第5章　N88-BASICからの移植

5.1　N88-BASICのプログラムを使う

「ぼくのプログラムはMS-DOS版じゃないから……」と、N88-BASICで入力したプログラムをそのまま眠らせていませんか。Quick-BASICでは、N88-BASICのプログラムを少し手直しするだけで走らせることができます。その場合もMS-DOSのシステムディスクをもっている人は、特別の出費も知識も必要としません。せっかく、PC-9801でプログラムを入力した人はその資産を。また、先輩諸氏が書いてくれたBASICの書籍をもっている人は、書籍の資産をそのまま生かしましょう。

1　N88-BASICはアスキーSAVE

アスキーSAVEとは、某出版社の「標準保存法」というのは、大ウソ。SAVEするときに、保存するファイル名の後ろにある"（ダブルクォーテーション）のさらにあとに、カンマを入れて、アルファベットのA（エイ）をつけるだけです。

```
SAVE "2:FILENAME.BAS",A [リターン]
```

この形で、プログラムをSAVEすると、テキストファイル形式で保存されます。長い間、プログラムを保存しっぱなしで、N88-BASICのコマンドを忘れている人がいらっしゃるかもしれませんので、ディスクのLOAD、SAVEについて少しだけ触れておきます。詳しくは、それぞれのマニュアルをご覧ください。

1．N88-BASICのシステムディスクをドライブ1（Aではありません）に入れて、N88-BASICをたち上げます。

2．プログラムを保存しているディスクを2（Bではありません）と仮定して、ディスクの中を見ます。

FILES 2 ［リターン］

```
Disk version
How many files(0-15)? 5
NEC N-88 BASIC(86) version 3.0
Copyright (C) 1983 by NEC Corporation / Microsoft Corp.
556140 Bytes free
Ok
FILES 2
menu   .    2    BUNSET SU  73    KANJI  DIC 5    format.nip 3    backup.n88 3
setinf.n88 1    xfiles.n88 1    sysgen.nip 2    format.hd  2    recov .hd  3
xfiles.hd  1    dir    .hd  1    backup.hd  2    mkfont.n88 2    switch.n88 4
dicmen.n88 5    DDconv.n88 1    mouse *cod 1    PAZ    .    1    CREATE.MJ  1
DATAIN.    2    SHUUKE.I1P 1    SHUUKE.I2P 1    SHUUKE.I3P 2    SHUUKE.I4P 2
SHUUKE.I5P 1    DATALI.ST  1    TESTPR.OG  1    TESTDI.G   1    OTOTES.T1  1
TAIKOU.3P  2    BUNSHO FMT 1               1    key pr.int 1    keypri nt  1
Ok

   load "  auto   go to   list   run        save "  key    print   edit . cont
```

3．ドライブ2の中の@@@@というプログラムファイルを、画面上に呼び出します。

LOAD "@@@@ ［リターン］

ファイル名の後ろの"は省略できます。

4．プログラムを走らせます。

```
RUN [リターン]
```

[f･5]を押してもプログラムを走らせることができます。

5．プログラムをアスキーファイル形式で保存します。このときに、拡張子「.BAS」を忘れないようにしてください。

```
SAVE"@@@@.BAS",A [リターン]
```

6．パソコンの電源を落として、N88-BASICを終了させます。
どうですか、思い出しました？

2 MS-DOSでプログラムファイルを変換

PC-9801がここまで普及したのは、当初のPC-8001の資産をPC-8801に、PC-8801の資産をPC-9801に受け継げるように設計されていだからだと、私は思っています。現に、8インチのディスクドライブは、88用のものを今も使っています。同様に、MS-DOSシステムの中には、FILECONV.EXEという、N88-BASICからMS-DOSへと、MS-DOSからN88-BASCへのファイル変換プログラムがあります。このプログラムを使って、N88-BASICで入力したプログラムリストをMS-DOSに変換します。

1．MS-DOSシステムディスクをAドライブに入れて、MS-DOSを起動させます。
2．最近のMS-DOSは、ディスクが何枚にも分かれていますので、FILECONV.EXEの入ったディスクを探して、Aドライブに入れてください。

```
FILECONV [リターン]
```

```
N88/MS-DOSファイルコンバータ (VER. 2.0)
Copyright (C) 1988 by NEC Corporation

------------------------------------------------------------------
変換：  1/N88-BASIC => MS-DOS    2/MS-DOS => N88-BASIC
------------------------------------------------------------------
選択して下さい
------------------------------------------------------------------
```

3．画面にN88-BASIC => MS-DOSが表示されます。AドライブにMS-DOSでフォーマットしたディスク、BドライブにN88-BASICのプログラムが入ったディスクを入れます。変換方法は、1/FILEを選択します。

```
------------------------------------------------------------------
変換方法は    1/FILE    2/VOLUME
------------------------------------------------------------------
選択して下さい
------------------------------------------------------------------
```

4．MS-DOS、N88-BASICのそれぞれのドライブをA:あるいはB:と指定すると、N88-BASICのディレクトリを表示します。ドライブ名の後ろの「:(コロン)」を忘れないで！　カーソル移動キーで、アスキーSAVEしたプログラムを選択します。

```
------------------------------------------------------------
N88-BASICのファイル名は SAMP.BAS
------------------------------------------------------------
選択して下さい
------------------------------------------------------------
```

5．MS-DOSのファイル名を入力すると、ファイルの変換が開始されます。

```
---------------[ N88-BASIC => MS-DOS ]---------------
SAMP.BAS    =>  SAMP.BAS
 JISコード変換
 KI/KOコード   DELETE




------------------------------------------------------------
変換は終了しました。別のファイルを変換しますか  1/YES    2/NO
------------------------------------------------------------
選択して下さい
------------------------------------------------------------
```

5.2 先輩の資産を生かす

本書では、わずかなプログラムリストしか掲載できませんので、『BASICで簡単にできるパソコン図形処理テクニック』（誠文堂新光社刊：赤松義幸著）という書籍の中のプログラムを使って、移植の方法を紹介します。

この本は、PC-8801/mkII、N88-BASIC用として、約330本の多彩な図形プログラムが掲載されており、そのプログラムの中から、1例をお借りして使用しています。
※本プログラムは、『パソコン図形処理テクニック』の112ページに掲載。

```
100 'QUAD.CO1 - ON QUADRATIC EQUATION
110 SCREEN 2:WIDTH 80,25:CONSOLE 0,25,0
120 PI=3.14159
130 INPUT "N";N
140 TH=PI*(.5-1/N):A=TAN(TH)
150 INPUT "B,C,L";B,C,L
160 IF B<L*A THEN PRINT "Can't !":GOTO 150
170 M=(A^2)/(4*(B-L*A))
180 FOR AA=0 TO 2*PI*(N-1)/N+.2 STEP 2*PI/N
190  FOR LL=-L TO L+4.9999 STEP 5
200   MX1=SQR((C-B)/M)+LL:MX2=-SQR((C-B)/M)+LL
210   FOR X=MX2 TO MX1 STEP (MX1-MX2)/20
220    Y=M*(X-LL)^2+B:GOSUB*ROTATION
230    IF X=MX2 THEN POINT(PX,PY):BX=X+(MX1-MX2):BY=Y
240    LINE-(PX,PY)
250   NEXT X:X=BX:Y=BY
260   GOSUB*ROTATION:LINE-(PX,PY)
270  NEXT LL
280 NEXT AA
290 END
300 *ROTATION
310 PX=X*COS(AA)-Y*SIN(AA)+320
320 PY=X*SIN(AA)+Y*COS(AA)+200
330 RETURN
```

N? 3
B,C,L? 30,150,30

N? 12
B,C,L? 100,180,20

1　赤松氏のプログラムの移植

MS-DOSのテキストファイルに変換したプログラムをQuick BASICに読み込んだら、インタプリタで実行します（[SHIFT]+[f･5]）。

最初に、サブルーチンのラベルがおかしいとメッセージが出ます。N88-BASICのラベルは、アステリクス＋ラベル名（*ラベル名）で、Quick BASICのラベル名は、ラベル名＋コロン（ラベル名:）です。

```
F/ファイル E/編集 V/表示 S/検索 R/実行 D/デバッグ C/関数 H/ヘルプ(f･1)
PR510.BAS
100 'QUAD.C01 - ON QUADRATIC EQUATION
110 SCREEN 2: WIDTH 80, 25: CONSOLE 0, 25, 0
120 PI = 3.14159
130 INPUT "N"; N
140 TH = PI * (.5 - 1 / N): A = TAN(TH)
150 INPUT "B,C,L"; B, C, L
160 IF B < L * A THEN PRINT "Can't !": GOTO 150
170 M = (A ^ 2) / (4 * (B - L * A))
180 FOR AA = 0 TO 2 * PI * (N - 1) / N + .2 STEP 2 * PI / N
190  FOR LL = -L TO L + 4.9999 STEP 5
200   MX1 = SQR((C - B) / M) + LL: MX2 = -SQR((C - B) / M) + LL
210   FOR X = MX2 TO MX1 STEP (MX1 - MX2) / 20
220    Y=M*(X-LL)^2+B:GOSUB*ROTATION
230    IF X=MX2 THEN POINT(PX,PY):BX=X+(MX1-MX2):BY=Y
240    LINE -(PX, PY)
250   NEXT X: X = BX: Y = B
260   GOSUB*ROTATION:LINE-(
270  NEXT LL

候補: ラベル・行番号
確認

メインプログラム: PR510.BAS   動作状況: 実行されていません   C 00013:028
```

プログラムの修正は全部で6カ所。行番はそのまま生かすことができます。また、プログラム中に使用されているGOTO文はそのままで有効です。

１．行番220の*ROTATION--------->ROTATION:

２．行番230のPOINT------------->PSET

３．行番260の*ROTATION--------->ROTATION:

４．行番300の*ROTATION--------->ROTATION:

５．行番110のCONSOLE 0,25,0---->削除

６．行番110のSCREEN 2---------->SCREEN 0

このうち、1．3．4．は、サブルーチンのラベル名の指定の仕方です。

2は、画面上に表示する点（ドット）位置の指定ですから、PSETを使います。

5は、画面の表示ずれを防ぐためのものです。Quick BASICでは必要ないので単純に削除しました。

6のSCREEN文は、N88-BASICとQuick BASICでは、引数が違います。正しい引数に訂正します。

修正といえるかどうかわかりませんが、プログラムを実行する前に画面消去のCLSを行番100の前に行番なしで追加しました。

2　赤松氏のプログラムの実行

原則として、元プログラムには手を加えないで、Quick BASICで走るように修正しました。いままで、触れていませんでしたが、Quick BASIC V4.2では、文字入力のテキスト画面を20行にすることができませんでした。V4.5からWIDTHを使って変更ができるようになっています。

※プログラムのアンダーラインの部分が、修正・追加・削除された場所です。

```
CLS
100 'QUAD.CO1 - ON QUADRATIC EQUATION
110 SCREEN 0: WIDTH 80, 25
120 PI = 3.14159
130 INPUT "N"; N
140 TH = PI * (.5 - 1 / N): A = TAN(TH)
150 INPUT "B,C,L"; B, C, L
160 IF B < L * A THEN PRINT "Can't !": GOTO 150
170 M = (A ^ 2) / (4 * (B - L * A))
180 FOR AA = 0 TO 2 * PI * (N - 1) / N + .2 STEP 2 * PI / N
190  FOR LL = -L TO L + 4.9999 STEP 5
200   MX1 = SQR((C - B) / M) + LL: MX2 = -SQR((C - B) / M) + LL
210   FOR X = MX2 TO MX1 STEP (MX1 - MX2) / 20
220    Y = M * (X - LL) ^ 2 + B: GOSUB ROTATION:
230    IF X = MX2 THEN PSET (PX, PY): BX = X + (MX1 - MX2): BY = Y
240    LINE -(PX, PY)
250   NEXT X: X = BX: Y = BY
```

```
260   GOSUB ROTATION: : LINE -(PX, PY)
270  NEXT LL
280 NEXT AA
290 END
300 ROTATION:
310 PX = X * COS(AA) - Y * SIN(AA) + 320
320 PY = X * SIN(AA) + Y * COS(AA) + 200
330 RETURN
```

3　元プログラムに加筆

プログラムの初期値入力のメッセージを日本語に書き換えて、係数ごとに数値を入力するように加筆しました。さらに、表示される線に色をつけます。

```
100 'QUAD.CO1 - ON QUADRATIC EQUATION
110 SCREEN 0: WIDTH 80, 25: CLS   'CLS追加
120 PI = 3.14159
130 INPUT "分割数＝", N
140 TH = PI * (.5 - 1 / N): A = TAN(TH)
150 INPUT "ユニット内部の長さ＝", B
    INPUT "ユニット外部の長さ＝", C
    INPUT "線の幅＝", L
160 IF B < L * A THEN COLOR 4: PRINT "内部と幅の数値が合わない！": COLOR 7: GOTO 150
170 M = (A ^ 2) / (4 * (B - L * A))
180 FOR AA = 0 TO 2 * PI * (N - 1) / N + .2 STEP 2 * PI / N
190  FOR LL = -L TO L + 4.9999 STEP 5
200   MX1 = SQR((C - B) / M) + LL: MX2 = -SQR((C - B) / M) + LL
210   FOR X = MX2 TO MX1 STEP (MX1 - MX2) / 20
220    Y = M * (X - LL) ^ 2 + B: GOSUB ROTATION:
       CL = RND(1) * 6 + 1
230    IF X = MX2 THEN PSET (PX, PY): BX = X + (MX1 - MX2): BY = Y
240    LINE -(PX, PY), CL
250   NEXT X: X = BX: Y = BY
```

```
260   GOSUB ROTATION: : LINE -(PX, PY), CL
270  NEXT LL
280 NEXT AA
290 END
300 ROTATION:
310 PX = X * COS(AA) - Y * SIN(AA) + 320
320 PY = X * SIN(AA) + Y * COS(AA) + 200
330 RETURN
```

5.3 未完の表計算プログラムの移植

このプログラムは、セル幅とセルの個数を自由に変更できるMS-DOS版のBASICプログラムです。このプログラムリストでは、画面スクロールを除いて、N88-BASICとQuick BASICの違いがすべて盛り込まれています。プログラムの修正個所にアンダーラインを入れておきました。よく見比べてください。

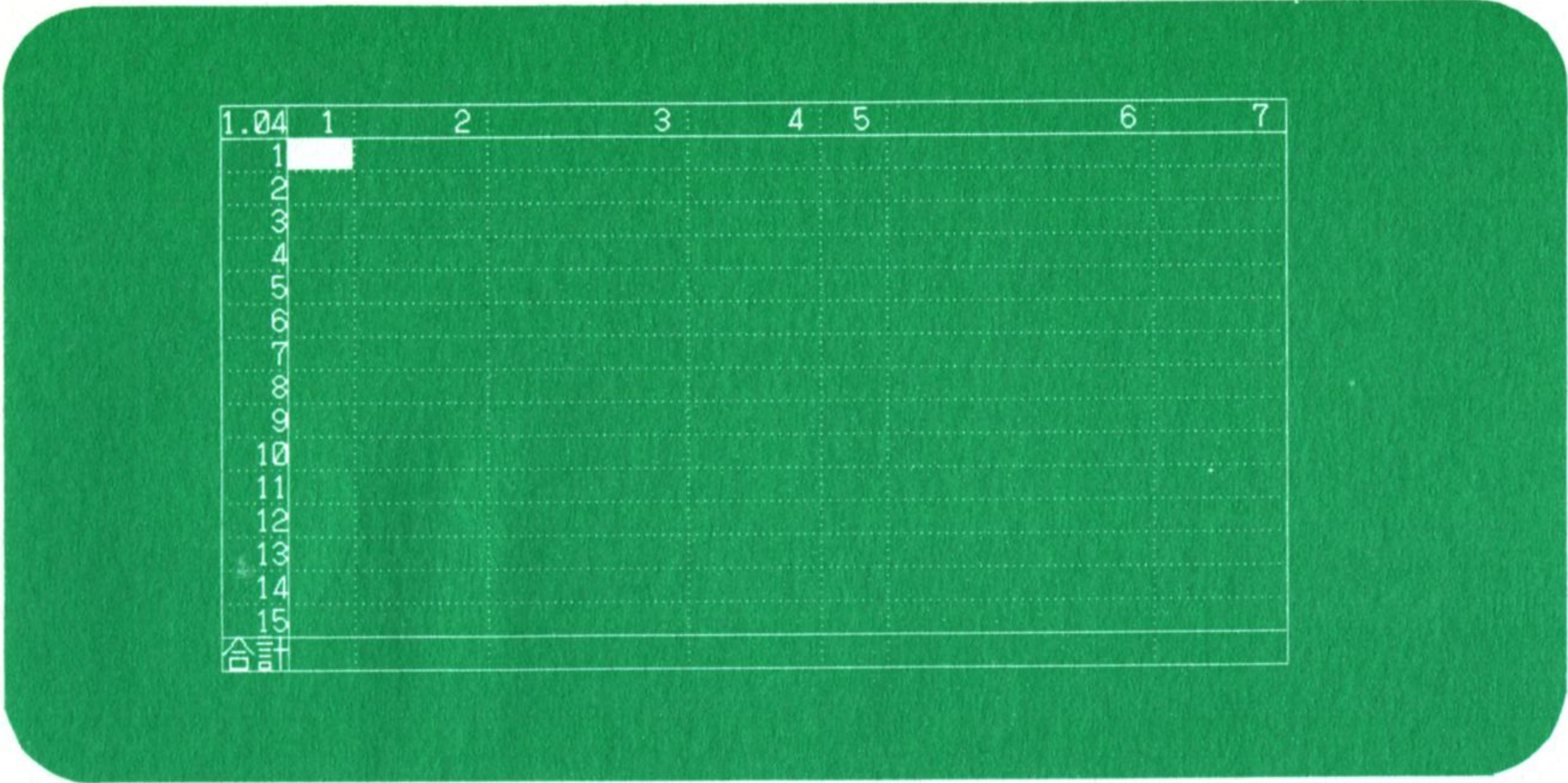
1　MS-DOS版BASICの表作成プログラム

このプログラムは、セル幅に合わせて、青いカーソルが移動できるようになっています。X方向のセルの数は、120行の変数X0で、Y方向のセルの数は変数Y0に代入します。また、セルに収めることのできる総文字数の指定は、全角文字数で、140行のDATA文の中に書き込んで指定します。このときのデータの数とX0で指定する数が一致する必要があります。DATAの数が少ないとエラーを起こすのは、N88-BASICでもQuick BASICでも同じです。このプログラムリストには、X方向、あるいはY方向の合計を出す計算式が入っていません。だから未完なのです。ただ、合計を出すためのセルに入力した数値データが、変数DAV(N,YN)に収められています。FOR...NEXTを使うと、合計を簡単に求めることができます。

グラフィックスの中に表を入れたのは、グラフィックスの線と矩形を使って、実用的なプログラムの参考になればと考えました。

```
10 '**********************************************************
20 '*          表作成プログラム                               *
30 '*                                    1990･2･13 (C)K.ENDOW  *
40 '**********************************************************
100 '画面制御と変数初期値の設定
110 CLS 3:SCREEN 3:WIDTH 80,25:CONSOLE ,,0,1            :'画面制御
120 X0=8:Y0=19:                                         'セルの数設定
130 DIM X(X0+1), U$(X0+1), DAV(X0,Y0):SUKU=16:X0=X0+1:Y0=Y0+2:DATA 2
140 DATA 2,4,6,4,2,8,4,4                               : 'セルに入れる文字数
150 FOR I=1 TO X0:READ X(I):XE=XE+X(I)
160 FOR J=1 TO X(I)*2:U$(I)=U$(I)+"#":NEXT J
170 NEXT:CLKX=4+X(2):CLKY=1
180 '
190 '
200 '表の作成開始
210 LINE(0,0)-(XE*SUKU,Y0*SUKU),4,B
220 FOR I=1 TO X0:X=X+X(I)
230 IF I=1 THEN LINE(X*SUKU,0)-(X*SUKU,Y0*SUKU),4 ELSE 240
240 LINE(X*SUKU,0)-(X*SUKU,Y0*SUKU),4,,&H8888
250 IF I=X0 THEN 270
260 LOCATE X*2,0:COLOR 7:PRINT USING U$(I+1);I;
270 NEXT:LOCATE 0,0:PRINT "1.04";
280 FOR Y=1 TO Y0
290 IF Y=1 OR Y=Y0-1 THEN LINE(0,Y*SUKU)-(XE*SUKU,Y*SUKU),4 ELSE 300
300 LINE(0,Y*SUKU)-(XE*SUKU,Y*SUKU),4,,&H8888
310 IF Y=Y0-1 THEN LOCATE 0,Y:PRINT "合計";:GOTO 340
320 IF Y=Y0 THEN 340
330 LOCATE 0,Y:PRINT USING"####";Y;
340 NEXT
350 '
360 '
400 'キー操作とデータ入力
410 N=1:YN=1:FL=1:GOSUB 610
420 DAT$=INKEY$:IF DAT$="" THEN 420
430 IF FL=1 AND DAT$=CHR$(&H1C) THEN GOSUB 660:GOTO 420'[→]キー
440 IF FL=1 AND DAT$=CHR$(&H1D) THEN GOSUB 700:GOTO 420'[←]キー
450 IF FL=1 AND DAT$=CHR$(&H1E) THEN GOSUB 740:GOTO 420'[↑]キー
```

```
460 IF FL=1 AND DAT$=CHR$(&H1F) THEN GOSUB 780:GOTO 420'[↓]キー
470 IF DAT$=CHR$(&HD) THEN FL=1:GOSUB 830:GOTO 420      '[CR]キー
480 IF DAT$<"0" OR DAT$>"9" THEN 420                '[0]~[9]までのキー
490 GOSUB 840:GOTO 420
500 '
510 '
600 '青色カーソルの表示と消去《サブ・ルーティン》
610 LINE(XN*16+33,YN*16+1)-((XN+X(N+1))*16+31,(YN+1)*16-1),1,BF:RETURN
620 LINE(XN*16+33,YN*16+1)-((XN+X(N+1))*16+31,(YN+1)*16-1),0,BF:RETURN
630 '
640 'カーソル位置の計算式《サブ・ルーティン》
650 'カーソル右への移動計算
660 GOSUB 620:N=N+1:XN=XN+X(N)
670 IF N>=X0 THEN N=1:XN=0:X(N)=X(N):BEEP
680 GOSUB 610:RETURN
690 'カーソル左への移動計算
700 GOSUB 620:N=N-1:XN=XN-X(N+1)
710 IF N<= 0 THEN XN=-2:N=X0-1:FOR JJ=2 TO X0:XN=XN+X(JJ-1):NEXT:BEEP
720 GOSUB 610:RETURN
730 'カーソル上への移動計算
740 GOSUB 620:YN=YN-1
750 IF YN<=0 THEN YN=Y0-2:BEEP
760 GOSUB 610:RETURN
770 'カーソル下への移動計算
780 GOSUB 620:YN=YN+1
790 IF YN>=Y0-1 THEN YN=1:BEEP
800 GOSUB 610:RETURN
810 '数値入力の決定と入力数値の表示《サブ・ルーチン》
830 DATT$="":GOSUB 660:RETURN
840 FL=0:DA$=RIGHT$(DAT$,1)
850 DATT$=DATT$+DA$
860 LOCATE XN*2+4,YN:PRINT DATT$:DAV(N,YN)=VAL(DATT$):RETURN
870 END
```

```
1.04 1      2        3      4  5          6       7
   1
   2
   3
   4
   5
   6
   7
   8
   9
  10
  11
  12
  13
  14
  15
合計
```

2　カーソル移動キーは、2バイトコード

N88-BASICからQuick BASICへの移植時の注意点は、ヘルプ画面[f･1]で見ることができます（画面はV4.2）。ただし、ここでは、カラーコードについては触れていません。第1章の図形を描くための基本的な命令を参照してください。

```
F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数    H/ヘルプ (f･1)
 ┌ SCREENモード                        ┌ デバイス名
 │81    テキストモード                 │KYBD: キーボード
 │84    テキスト+グラフィックス(640×200)│SCRN: 画面
 │87    グラフィックモード     (640×400)│COMn: RS-232Cポート
 │88=0  テキスト+グラフィックス(640×400)│LPTn: プリンタポート
                                       │CONS: 標準出力
 ┌ N88BASICから QuickBASICへの移植時の注意点
                     N88BASIC             QuickBASIC 4.2
 │BEEPの引数         BEEP [0:1]           BEEP (引数なし)
 │スクロール範囲     CONSOLE Y1, Y2       VIEW PRINT Y1+1 TO Y2+1
 │LOCATEの引数       LOCATE X, Y          LOCATE Y+1 , X+1
 │WIDTHの引数        80:40, 25:20         80, 25 のみ
 │ラベル名           *labelname           labelname:

 N88BASIC で ASCIIセーブ(SAVE "Flename",A) したものを QuickBASICで
 読み込み修正して下さい. 行番号は取らなくても実行可能です.

   N/次のページ     P/前のページ     K/キーワード     取消

メインプログラム: PR502.BAS     動作状況: PR502.BAS          C  00001:005
```

▲N88BASICとは、MS-DOS版のことです（V4.2のヘルプ画面）

▼特殊キーは、2バイトコードになっています（V4.2のヘルプ画面）

```
F/ファイル  E/編集  V/表示  S/検索  R/実行  D/デバッグ  C/関数     H/ヘルプ(f･1)

10   特殊キーの INKEY$ による戻り値(16進)            特殊キーの INKEY$による
20   INS      00+52      f･1       00+3B            戻り値は 2バイトの文字で
30   DEL      00+53      f･2       00+3C            返されます.
40   UP       00+48      f･3       00+3D            例えば HOME キーが押され
100  LEFT     00+4B      f･4       00+3E            たことをチェックする場合,
110  RIGHT    00+4D      f･5       00+3F            以下のようにします.
120  DOWN     00+50      f･6       00+40            A$=INKEY$
130  HOME     00+47      f･7       00+41            IF A$=CHR$(00,&H47) THEN
140  HELP     00+4F      f･8       00+42               ...
150  ROLLUP   00+51      f･9       00+43            END IF
160  ROLLDOWN 00+49      f･10      00+44
170
180  ON KEY (n) の割り込みキー                      ユーザ定義KEYフラグ(16進)
190  1-10   f･1-f･10   13     RIGHT                 00 Normal     04 カナ
200  11     UP         14     DOWN                  01 SHIFT      08 GRPH
210  12     LEFT       15-25  ユーザ定義            02 CAPS       10 CTRL
220
230

     N/次のページ    P/前のページ    K/キーワード    取消

メインプログラム: PR502.BAS     動作状況: PR502.BAS              C  00001:005
```

Quick BASICへの移植のときに、[HOME]キーを押すと、計算サブルーチンへジャンプするように、プログラムが書き加えられています。ヘルプ画面で特殊キーの戻り値は、2バイト文字で返されると書いてあります。

2バイト文字の表示の仕方は、次のような方法が使えます。

HELP画面　IF A$=CHR$(00,&H47)

◎　IF A$=CHR$(0,&H47)

　　Quick BASICでは、00と入力しても0となります。

◎　IF A$=CHR$(&H0,&H47)

◎　IF A$=CHR$(&H00)+(&H47)

```
10 '*************************************************************
20 '*          表作成プログラム                                 *
30 '*                              1990･1･13 (C)K.ENDOW         *
40 '*************************************************************
100 '画面制御と変数初期値の設定
110 CLS : SCREEN 0: WIDTH 80, 25:                    '画面制御
120 X0 = 7: Y0 = 16: CEL = 2                         'セルの数設定
130 DIM X(X0 + 1), U$(X0 + 1), DAV(X0 + 1, Y0 + 1): SUKU = 16: X0 = X0 + 1: Y0 = Y0 + 1: DATA 2
```

```
140 DATA 2,4,6,4,2,8,4,4                              : 'セルに入れる文字数
150 FOR I = 1 TO X0: READ X(I): XE = XE + X(I)
160 FOR J = 1 TO X(I) * 2: U$(I) = U$(I) + "#": NEXT J
170 NEXT: CLKX = 4 + X(2): CLKY = 1
200 '表の作成開始
210 LINE (0, 0)-(XE * SUKU, Y0 * SUKU), CEL, B
220 FOR I = 1 TO X0: X = X + X(I)
230 IF I = 1 THEN LINE (X * SUKU, 0)-(X * SUKU, Y0 * SUKU), CEL ELSE 240
240 LINE (X * SUKU, 0)-(X * SUKU, Y0 * SUKU), CEL, , &H8888
250 IF I = X0 THEN 270
260 LOCATE 1, X * 2: COLOR 7: PRINT USING U$(I + 1); I;
270 NEXT: LOCATE 1, 1: PRINT "1.04";
280 FOR y = 1 TO Y0
290 IF y = 1 OR y = Y0 - 1 THEN LINE (0, y * SUKU)-(XE * SUKU, y *
SUKU), CEL ELSE 300
300 LINE (0, y * SUKU)-(XE * SUKU, y * SUKU), CEL, , &H8888
310 IF y = Y0 THEN LOCATE y, 1: PRINT "合計"; : GOTO 340
320 IF y = Y0 THEN 340
330 LOCATE y + 1, 1: PRINT USING "####"; y;
340 NEXT
400 'キー操作とデータ入力
410 N = 1: YN = 1: FL = 1: GOSUB 610
420 DAT$ = INKEY$: IF DAT$ = "" THEN 420
    IF FL = 1 AND DAT$ = CHR$(0, &H47) THEN GOSUB KEISAN: 'HOME
430 IF FL = 1 AND DAT$ = CHR$(0, &H4D) THEN GOSUB 660: GOTO 420'[→]キー
440 IF FL = 1 AND DAT$ = CHR$(0, &H4B) THEN GOSUB 700: GOTO 420'[←]キー
450 IF FL = 1 AND DAT$ = CHR$(0, &H48) THEN GOSUB 740: GOTO 420'[↑]キー
460 IF FL = 1 AND DAT$ = CHR$(0, &H50) THEN GOSUB 780: GOTO 420'[↓]キー
470 IF DAT$ = CHR$(&HD) THEN FL = 1: GOSUB 830: GOSUB SYOKYO: : GOTO
420'[CR]キー
480 IF DAT$ < "0" OR DAT$ > "9" THEN 420              '[0]～[9]までのキー
490 GOSUB 840: GOTO 420
600 '青色カーソルの表示と消去《サブ・ルーティン》
610 LINE (XN * 16 + 33, YN * 16 + 1)-((XN + X(N + 1)) * 16 + 31, (YN +
1) * 16 - 1), 1, BF: RETURN
620 LINE (XN * 16 + 33, YN * 16 + 1)-((XN + X(N + 1)) * 16 + 31, (YN +
1) * 16 - 1), 0, BF: RETURN
```

```
630 '
640 'カーソル位置の計算式《サブ・ルーティン》
650 'カーソル右への移動計算
660 GOSUB 620: N = N + 1: XN = XN + X(N)
670 IF N >= X0 THEN N = 1: XN = 0: X(N) = X(N): BEEP
680 GOSUB 610: RETURN
690 'カーソル左への移動計算
700 GOSUB 620: N = N - 1: XN = XN - X(N + 1)
710 IF N <= 0 THEN XN = -2: N = X0 - 1: FOR JJ = 2 TO X0: XN = XN + X(JJ
    - 1): NEXT: BEEP
720 GOSUB 610: RETURN
730 'カーソル上への移動計算
740 GOSUB 620: YN = YN - 1
750 IF YN <= 0 THEN YN = Y0 - 2: BEEP
760 GOSUB 610: RETURN
770 'カーソル下への移動計算
780 GOSUB 620: YN = YN + 1
790 IF YN >= Y0 - 1 THEN YN = 1: BEEP
800 GOSUB 610: RETURN
810 '数値入力の決定と入力数値の表示《サブ・ルーティン》
830 DATT$ = "": GOSUB 660: RETURN
840 FL = 0: DA$ = RIGHT$(DAT$, 1)
850 DATT$ = DATT$ + DA$
860 LOCATE YN + 1, XN * 2 + 5: PRINT DATT$: DAV(N, YN) = VAL(DATT$):
RETURN
870 END
KEISAN:
  :
  LOCATE Y0 + 2, 5: PRINT "計算サブルーチン"
  :
 RETURN
SYOKYO:
  :
  LOCATE Y0 + 2, 5: PRINT "                "
  :
 RETURN
```

3　特殊キーを使った、模擬カーソルの移動

カーソルの形は、"★"を使います。カーソルの1文字ずつの上下左右の移動、カーソル移動キーを使います。カーソルを画面の中央、白いマークの位置は通過でき、そのほかの壁では通過できません。ただし、[HELP]キーを押すと通過できるというのはどうでしょう。

```
'PR520.BAS
SCREEN 0: CLS
FOR X1 = 0 TO 480 STEP 32
    LINE (X1, 0)-(X1 + 32, 16), 7, B
     PAINT (X1 + 1, 8), CHR$(&H2) + CHR$(&H0) + CHR$(&H77), 7
      LINE (X1, 0)-(X1 + 32, 16), 0, B
      LINE (X1, 320)-(X1 + 32, 336), 7, B
     PAINT (X1 + 1, 321), CHR$(&H2) + CHR$(&H0) + CHR$(&H77), 7
    LINE (X1, 320)-(X1 + 32, 336), 0, B
NEXT
FOR Y1 = 16 TO 304 STEP 16
    LINE (0, Y1)-(24, Y1 + 16), 7, B
     PAINT (2, Y1 + 1), CHR$(&H2) + CHR$(&H0) + CHR$(&H77), 7
      LINE (0, Y1)-(24, Y1 + 16), 0, B
      LINE (488, Y1)-(512, Y1 + 16), 7, B
     PAINT (489, Y1 + 1), CHR$(&H2) + CHR$(&H0) + CHR$(&H77), 7
    LINE (488, Y1)-(512, Y1 + 16), 0, B
NEXT
       LINE (8 * 29, 10)-(8 * 31, 16), 7, BF
       LINE (8 * 29, 320)-(8 * 31, 326), 7, BF
       LINE (18, 16 * 10)-(24, 16 * 11), 7, BF
       LINE (489, 16 * 10)-(496, 16 * 11), 7, BF
    CXS = 4: CXE = 60: CX = 30: CYS = 2: CYE = 20: CY = 10
    ST$ = "★": SB$ = "  "
     LOCATE CY, CX: PRINT ST$;
     LOCATE 1, 66: PRINT "ワープゾーン"
     LOCATE 5, 72: PRINT "上"
     LOCATE 6, 72: PRINT "↑"
```

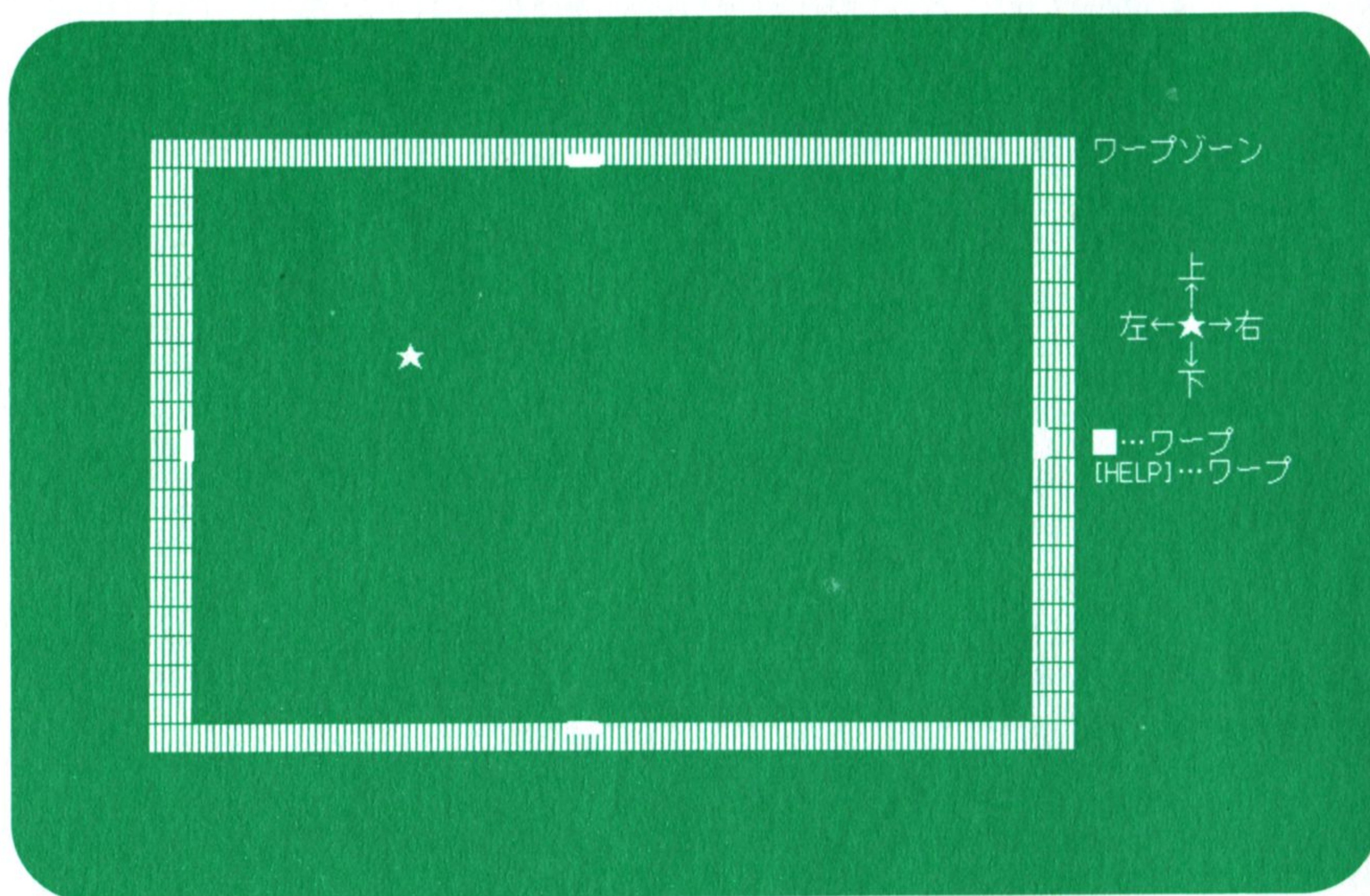
```
      LOCATE 7, 68: PRINT "左←★→右"
      LOCATE 8, 72: PRINT "↓"
      LOCATE 9, 72: PRINT "下"
      LOCATE 11, 66: PRINT "■…ワープ"
      LOCATE 12, 66: PRINT "[HELP]…ワープ"
DO
     ANS$ = INKEY$
      IF ANS$ = CHR$(0, &H4F) THEN TF = 1
      IF ANS$ = CHR$(0, &H48) THEN GOSUB UP:
      IF ANS$ = CHR$(0, &H50) THEN GOSUB DOWN:
      IF ANS$ = CHR$(0, &H4D) THEN GOSUB RIGHT:
      IF ANS$ = CHR$(0, &H4B) THEN GOSUB LEFT:
      IF ANS$ = "E" THEN END
      LOCATE CY, CX: PRINT ST$;
LOOP
END
UP:
   GOSUB STB:
   CY = CY - 1
   IF CY < CYS AND (TF = 1 OR CX = 30) THEN
     CY = CYE
   ELSEIF CY < CYS AND TF = 0 THEN
     BEEP: CY = CYS
   END IF
   TF = 0
   RETURN
DOWN:
   GOSUB STB:
   CY = CY + 1
   IF CY > CYE AND (TF = 1 OR CX = 30) THEN
     CY = CYS
   ELSEIF CY > CYE AND TF = 0 THEN
     BEEP: CY = CYE
   END IF
   TF = 0
   RETURN
RIGHT:
```

```
    GOSUB STB:
    CX = CX + 1
    IF CX > CXE AND (TF = 1 OR CY = 11) THEN
      CX = CXS
    ELSEIF CX > CXE AND TF = 0 THEN
      BEEP: CX = CXE
    END IF
     TF = 0
    RETURN
LEFT:
    GOSUB STB:
    CX = CX - 1
    IF CX < CXS AND (TF = 1 OR CY = 11) THEN
      CX = CXE
    ELSEIF CX < CXS AND TF = 0 THEN
      BEEP: CX = CXS
    END IF
     TF = 0
    RETURN
STB:
    LOCATE CY, CX: PRINT SB$
    RETURN

KEISAN:
                    M = VAL(RIGHT$(TIME$, 2))
                S = VAL(MID$(TIME$, 4, 2))
            T = VAL(LEFT$(TIME$, 2))
     IF T >= 13 THEN T = T - 12
            TT = T * 5 + S / 12
                    LINE (320, 200)-(XX(M), YY(M)), 7
                LINE (320, 200)-(XX(S), YY(S)), 6

            LINE (320, 200)-(XX(TT), YY(TT)), 4
                    LOCATE 15, 37: PRINT TIME$
                LINE (35 * 8, 14 * 16)-(45 * 8, 15 * 16), 7, B, &H8888
                    LINE (320, 200)-(XX(M), YY(M)), 0
                LINE (320, 200)-(XX(S), YY(S)), 0
```

```
        LINE (320, 200)-(XX(TT), YY(TT)), 0
            CIRCLE (320, 200), 4, 4
RETURN
END
```

ワープゾーン

上
左←★→右
下

■…ワープ
[HELP]…ワープ

Quick BASIC

付　　録

付録 1　*Quick BASIC V4.2*のセットアップ

1.1　Quick BASICに入っているファイル

Quick BASICには、2枚のフロッピーディスクがハードケースに入っています。1枚には、『プログラムディスク』と書かれており、もう1枚のほうのフロッピーディスクには、『ライブラリ&サンプルプログラム』と書かれています。それぞれのディスクには、次のようなファイルが収納されています。

1　プログラムディスク

プログラムディスクの中は、3つの階層ディレクトリと5つのファイルからなっています。ファイルのタイムスタンプは、すべてのファイルが4:20になっています。これは、Quick BASICのバージョンを表わしています。

```
A>DIR

 ドライブ A: のディスクのボリュームラベルは QB42_N01
 ディレクトリは A:¥

QB_ENV        <DIR>      88-12-15    4:20
QB_CMP        <DIR>      88-12-15    4:20
LIB_E         <DIR>      88-12-15    4:20
SETUP    DOC      2763   88-12-15    4:20
SETUP    EXE     30198   88-12-15    4:20
PACKING  LST      3303   89-01-15    4:20
README   DOC      7832   89-01-15    4:20
SAMPLE   DOC      4196   89-01-15    4:20
         8 個のファイルがあります.
   176128 バイトが使用可能です.

A>

  C1   CU   CA   S1   SU     VOID  NWL  INS  REP  ^Z
```

QB_ENV	<DIR>	ディレクトリ(統合環境)
MOUSE.COM		マウスドライバ(常駐型式)
MOUSE.SYS		マウスドライバ
QB.BI		QuickBASICヘッダーファイル
QB.EXE		QuickBASIC本体
QB.HLP		ヘルプファイル
QB.INI		イニシャルファイル
QB.PIF		MS-WINDOWS 2.0 用 Picture File
QB_CMP	<DIR>	ディレクトリ(コンパイラ)
BC.EXE		BASIC Compiler
BRUN42A.EXE		ランタイムモジュール(FPA)
BRUN42E.EXE		ランタイムモジュール(FPI)
LIB.EXE		ライブラリアン
LINK.EXE		リンカ
LIB_E	<DIR>	ディレクトリ(FPIライブラリ)
BCOM42E.LIB		スタンドアロンライブラリ(FPI)
BRUN42E.LIB		分離型ライブラリ(FPI)
SETUP.DOC		*セットアップの説明
SETUP.EXE		セットアッププログラム
PACKING.LST		*パッキングリスト
README.DOC		*マニュアルの追加付加情報
SAMPLE.DOC		*サンプルプログラムの説明

*印のファイルは、TYPE ファイル名.拡張子［リターン］で内容を見ることができます。

2 ライブラリ&サンプルプログラムディスク

ライブラリとサンプルプログラムが入っています。

```
A>DIR

 ドライブ A: のディスクのボリュームラベルは QB42_NO2
 ディレクトリは A:¥

LIB_A        <DIR>     88-12-15    4:20
SOURCE       <DIR>     88-12-15    4:20
        2 個のファイルがあります.
   818176 バイトが使用可能です.

A>

 C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

LIB_A	<DIR>	ディレクトリ(FPAライブラリ)
BCOM42A.LIB		スタンドアロンライブラリ(FPA)
BQLB42.LIB		QLB 構築ライブラリ
BRUN42A.LIB		分離型ライブラリ(FPA)
QB.LIB		ライブラリ
QB.QLB		Quickライブラリ
SOURCE	<DIR>	ディレクトリ(サンプルソース)
BAR.BAS		棒グラフ
CAL.BAS		カレンダー
CCALL.C		Cの呼び出し
CCALL.QLB		Cの呼び出し
CCALL.BAS		Cの呼び出し
COLOR.BAS		色の表示
CRLF.BAS		フィルター
CUBE.BAS		キューブ

DISPLAY.BAS	ファイルのタイプ
DRAW.BAS	お絵書き
EDIT.BAS	文字入力&FEP ON/OFF
EDIT.BI	EDIT.BAS 用のヘッダーファイル
EDPAT.BAS	タイルパターンエディタ
GET.BAS	PUT/GETの例
HANOI.BAS	ハノイの塔
INDEX.BAS	簡易データベース
MANDEL.BAS	マンデルブロートセット
PALETTE.BAS	パレット文の例
PSETBALL.BAS	Ball の跳ね返り
QBL.BAS	コンパイラドライバ(コンパイル後利用)
QLBDUMP.BAS	QLB内容表示
SEARCH.BAS	文字検索
SIERP.BAS	シェルピンスキー曲線
SINWAVE.BAS	サインウェーブ
STRTONUM.BAS	数値変換
STYLE.BAS	ラインスタイル
TERMINAL.BAS	ターミナルソフト
WHEREIS.BAS	ファイル検索
ABSOLUTE.ASM	アセンブリソース
INTRPT.ASM	アセンブリソース
GCOPY.BAS	グラフィック画面印字プログラム

1.2 Quick BASICのワークディスクを作る

Quick BASICのセットアップユーティリティ（SETUP.EXE）を使えば、自分のディスクにワークディスクを作ることができます。そのためには、いくつかの前準備が必要です。

1　MS-DOSでフォーマット

Quick BASICを動かすためには、新しく用意したフロッピーディスクに、MS-DOSのシステムとQuick BASICのシステムを組み込む必要があります。この、新しいシステムのディスクを作る作業をセットアップ、またはインストールといいます。本書では、セットアップという言葉を使ってきました。

なお、使用しているディスクの種類（1MB、640KBのフロッピーディスク、ハードディスクなど）によって、組み込めるファイルの数やディレクトリの作り方が変わってきます。本書では、PC-9801Vm2を使っている関係から、1MBのフロッピーディスクを中心に話を進めます。

まず、これから自分が使用する新しいフロッピーディスクにMS-DOSのシステムを乗せます。そのために、フォーマットとシステムのコピーを行ないます。これを一度に行なってくれるMS-DOSのコマンドがFORMAT/Sです。

それでは、MS-DOSを起動させたあと、画面にA>（エープロンプト）が表示された状態にします。画面に、MS-DOSのメニュー画面が出ているときには、[STOP]キーを押して、A>の状態にします。

ドライブBに、これから使う新しいフロッピーディスクを入れます。

```
A>FORMAT/S B:［リターン］
```

2　MS-DOSの外部コマンドのコピー

MS-DOSの外部コマンドで必要なものをコピーしておきます。まず、フォーマット用のFORMAT.EXE（MS-DOSのバージョンによっては、FORMAT.COM）です。本書で使用しいるMS-DOS Ver.3.3は、PRINT.SYSが必要なので、これも同時にコピーします。MS-DOSのバージョンによって、PRINT.SYSを必要としないものもあります。

```
A>COPY A:FORMAT.EXE B:
        1 個のファイルをコピーしました.

A>COPY A:PRINT.SYS B:
        1 個のファイルをコピーしました.

A>

C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

▲COPY A:FORMAT.EXE B:とCOPY A:PRINT.SYSの画面

1.3 セットアップ

Quick BASICのセットアップを行ないます。まず、ドライブAに、Quick BASICのプログラムディスクを入れます。

1 「セットアップユーティリティ」の起動

起動の仕方は、

SETUP ［リターン］

です。

最初に、セットアップユーティリティの説明が、3画面分表示されます。それぞれの画面を読み終わったら、[STOP]キーと[ESC]キー以外の何かのキーを押します。習慣として、[スペース]キーか、［リターン］キーを押すようにしましょう。

```
QuickBASIC セットアップ ユーティリティ        Ver1.00

---準備---                                                    (1/3)
◎フロッピーディスクへセットアップする場合、フォーマット済みのディスクを用意し
  てください。

  [フロッピーベースでQuickBASICを利用する方へ]
  あらかじめフォーマット済みのディスクが必要です。
  メディアのタイプによって必要なディスクの枚数が異なります。
  ワークディスクには FORMAT /S スイッチでMS-DOSシステムを転送しておいて
  下さい。システムが転送されていない場合はセットアップはおこなわれません。
  ・HDタイプ(3.5 2HD, 5 2HD)の場合
      1枚：ワークディスク
  ・DDタイプ(3.5 2DD, 5 2DD)の場合
      2枚：ワークディスク、コンパイラディスク
      (FORMAT /9 で 720Kフォーマットにしておく方が良いでしょう。)
◎フォーマットが済みましたら添付のシール(「ワークディスク」、「コンパイラディ
  スク」)をディスクに貼っておいて下さい。他のシールは使いません。御自由にお使い
  下さい。
フォーマット済みのディスクが用意していなければ、CTRL+C でセットアップを中止して
ディスクのフォーマットを行って下さい。

【何かキーを押して下さい】

  C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

```
QuickBASIC セットアップ ユーティリティ        Ver1.00

---サンプルプログラム---                                        (2/3)
セットアップでは、サンプルプログラムを転送しません。必要に応じて転送をおこなっ
て下さい。

---ライブラリについて---
QuickBASICのコンパイラ(BC.EXE)が通常利用するライブラリは、以下のもの
です。
BRUN42A.EXE  ランタイムモジュール
BRUN42A.LIB  ランタイムライブラリ
BCOM42A.LIB  スタンドアロンライブラリ

これらを代替数値演算ライブラリ(ALT-MATH)と呼びます。これらは、数値演算コプロセ
ッサの有無に関わらず、一定の速度で高精度な計算を実現します。
数値演算コプロセッサを装着している方のため BRUN42E.EXE,BRUN42E.LIB,BCOM42E.LIB
(80?87エミュレートライブラリ)を用意しています。これによってより高速な演算を実現
できます。
フロッピーベースでご利用になる場合は、セットアップではエミュレートライブラリは
転送できません。必要な場合は、マスターディスクから転送して下さい。

【何かキーを押して下さい】

  C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

QuickBASIC セットアップ ユーティリティ　　　Ver1.00

---フロントプロセッサの組み込み---　(3/3)

QuickBASICに日本語入力システムを組み込む場合は、お使いのフロントプロセッサのマニュアルを参照して各自おこなってください。
なお、ご利用可能なフロントプロセッサは、ATOK6、MS-KANJI APIのものです。添付の「日本語入力システムのご使用について」を参考にしてください。

---CONFIG.SYS、AUTOEXEC.BAT---

このセットアッププログラムは、フロッピーへセットアップしたとき「ワークディスク」に CONFIG.SYS , AUTOEXEC.BAT を作成します。ハードディスクの場合は、指定されたディレクトリに、NEW-CONF.SYS, NEW-PATH.BAT を作成しますので、これらのファイルを参考にして CONFIG.SYS AUTOEXEC.BAT を修正してください。環境変数については、README.DOC をご参照下さい。

なお、README.DOC には補足の情報や、訂正資料が記載されています。必ずお読み頂くようお願いします。

【何かキーを押して下さい】

C1 CU CA S1 SU VOID NWL INS REP ^Z

いよいよ、セットアップの開始です。画面上にメッセージが表示され、質問にYes、Noで、答えていくと自動的にセットアップが行なわれます。

QuickBASIC セットアップ ユーティリティ　　　Ver1.00

●セットアップを開始しますか [Y/N] ? Y

これより、QuickBASICのセットアップを開始します。
もし、セットアップの途中でCTRL+CもしくはSTOPキーを押した場合、セットアップ作業を中断します。
その場合は、もう一度 SETUP コマンドを実行して、最初から設定をやり直して下さい。また、セットアップが終了した後に設定を変更したい場合も、もう一度 SETUP コマンドを実行し最初から設定をやり直して下さい。

C1 CU CA S1 SU VOID NWL INS REP ^Z

［Y］、［N］のあとは、ドライブBにファイルのコピーが開始されます。途中でフロッピーディスクを抜かないように注意してください。

```
QuickBASIC セットアップ ユーティリティ        Ver1.00

●ハードディスクにセットアップしますか [Y/N] ? N
●QuickBASICをどのドライブにインストールしますか? B
●マウスを使用しますか [Y/N] ? Y
●マウスの種類を選択して下さい [1:バス/2:シリアル] ? 1
●マウスドライバを選択して下さい [1:MOUSE.SYS/2:MOUSE.COM] ? 2


上記の設定でよろしいですか [Y/N]?Y
------------------------------------------------------------
ドライブ B にワークディスクをセットして下さい。
【何かキーを押して下さい】

作業中です。 しばらくお待ち下さい。


A:¥QB_ENV¥QB.EXE --->B:¥BIN¥QB.EXE

  C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

続いて、ライブラリディスクをAドライブにセットするように指示があります。ディスクを差し替えたら、［STOP］キー以外の「何かのキー」を押します。

```
QuickBASIC セットアップ ユーティリティ        Ver1.00

●ハードディスクにセットアップしますか [Y/N] ? N
●QuickBASICをどのドライブにインストールしますか? B
●マウスを使用しますか [Y/N] ? Y
●マウスの種類を選択して下さい [1:バス/2:シリアル] ? 1
●マウスドライバを選択して下さい [1:MOUSE.SYS/2:MOUSE.COM] ? 2


上記の設定でよろしいですか [Y/N]?Y
------------------------------------------------------------
ドライブ A にライブラリディスクをセットして下さい。
【何かキーを押して下さい】



  C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

ライブラリディスクのファイルのコピーが終了すると、セットアップが自動的に終了して、メッセージが表示されます。

```
以上でセットアップを終了します。

【ご注意】
ＱｕｉｃｋＢＡＳＩＣ使用時の注意事項に関しては、ＲＥＡＤＭＥ．ＤＯＣに
記載されています。ＱｕｉｃｋＢＡＳＩＣ起動前に必ずご一読下さい。
「ワークディスク」（ハードディスクの場合はセットアップしたディレクトリ
（デフォルトでは￥ＱＢ））内のＲＥＡＤＭＥ．ＤＯＣに書かれています。
ＲＥＡＤＭＥ．ＤＯＣファイルの参照方法は以下の通りです。
《フロッピーディスクの場合》
ＭＳ－ＤＯＳ起動後に「ワークディスク」をＡドライブに入れ、以下を実行して
下さい。
        A> TYPE README.DOC
《固定ディスクの場合》
ＭＳ－ＤＯＳ起動後にＱｕｉｃｋＢＡＳＩＣをセットアップしたディレクトリに
移動して以下を実行して下さい。
        A> TYPE README.DOC

A>

   C1   CU   CA   S1   SU     VOID  NWL  INS  REP  ^Z
```

以上で、セットアップは終了です。画面上で、README.DOCを見るように注意が表示されています。［CTRL］キーを押したまま［S］キーを押すとREADME.DOCの画面を停止させることができます。再スタートは「適当なキー」を押してください。再び停止させるときは、［CTRL］＋［S］キーです。

3　ワークディスクの内容

ワークディスクの内容を見ると、次頁のような構成になっています。

ファイル名の頭にQBのついたファイルはインタープリタ用、BCがついているファイルはコンパイラに必要なファイルです。

```
A>DIR B:

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM    24931  88-07-13    0:00
BIN           <DIR>    90-05-04   11:51
LIB           <DIR>    90-05-04   11:51
README   DOC     7832  89-01-15    4:20
AUTOEXEC BAT       68  90-05-04   11:55
CONFIG   SYS       22  90-05-04   11:55
FORMAT   EXE    97766  88-07-13    0:00
PRINT    SYS     5855  88-07-13    0:00
        8 個のファイルがあります.
    52224 バイトが使用可能です.

A>

    C1  CU  CA  S1  SU    VOID  NWL  INS  REP  ^Z
```

BIN <DIR>			
QB.EXE	QuickBASICインタープリタ		統合
QB.HLP	QuickBASICヘルプファイル		統合
QB.INI	※ QuickBASIC画面情報		統合
QB.BI	※ QB.QLB用インクルードファイル	SOURCE	統合
MAUSE.COM	マウスドライバー		
BC.EXE	ベーシックコンパイラ		コンパイラ
LIB.EXE	ライブラリアン		コンパイラ
LINK.EXE	リンカ		コンパイラ
BRUN42A.EXE	ランタイムモジュール(FPA)		コンパイラ
LIB <DIR>			
BCOM42A.LIB	スタンドアロンライブラリ(ALT-MATH)		コンパイラ
	スタンドアロンEXEファイルを作成時に必要		
BRUN42A.LIB	ランタイムライブラリ(ALT-MATH)		コンパイラ
BQLB42.LIB	QLB 構築ライブラリ		コンパイラ
QB.LIB	※ デフォルトライブラリ		

```
  QB.QLB        ※  デフォルトQuickライブラリ

COMMAND.COM         MS-DOSのシステム      本書は、Ver.3.3使用
FORMAT.EXE          MS-DOSの外部コマンド
PRINT.SYS           プリンタドライバ
README.DOC      ※  マニュアルの追加付加情報
CONFIG.SYS          システムのコンフィギュレーションファイル
AUTOEXEC.BAT        プログラム自動実行用バッチファイル
```

※印のファイルは、Quick BASICの作動には、直接関係ありません。フロッピーディスクの容量不足で、緊急かつ、ニッチもサッチもいかなくなったときには削除できます。

4 CONFIG.SYSとAUTOEXEC.BAT

セットアップの終了したファイルを見ると、新たにCONFIG.SYSとAUTOEXEC.BATというファイルが増えています。

CONFIG.SYSは、MS-DOSを起動したときに、ルートディレクトリにあると、ファイルの内容にしたがって、デバイスドライバの登録やバッファの大きさの設定などのシステムを構築してくれます。

簡単にいうと、ディスクドライブやプリンタ、画面などとのやり取りを設定するファイルと考えてください。セットアップするときの条件で変わってきますが、マニュアルどおりにセットアップを行なうと、次のように設定されています。CONFIG.SYSの内容を見るときには、MS-DOSの内部コマンドTYPEを使います。

また、本書で使っているMS-DOSの場合は、PRINT.SYSを使うので、CONFIG.SYSに追加して書き加えます。CONFIG.SYSの書き換えには、COPYコマンドを使います。

1行の入力が終了すると［リターン］キーを押して改行します。

```
A>TYPE CONFIG.SYS
FILES=20
BUFFERS=20

A>COPY CON CONFI.SYS
FILES=20
BUFFERS=20
DEVICE=A:¥PRINT.SYS
^Z
        1 個のファイルをコピーしました.

A>
```

最後に、[CTRL]キーを押しながら、[Z]キーを押すと、CONFIG.SYSの書き換えが終了します。

また、AUTOEXEC.BATはプログラムを自動実行するバッチファイルです。中身は、次のようになっています。ここの段階では、書き換え作業は行ないません。

```
A>TYPE AUTOEXEC.BAT
SET COMSPEC=A:¥COMMAND.COM
SET PATH=A:¥BIN
SET LIB=A:¥LIB
MOUSE

A>







 C1   CU   CA   S1   SU    VOID  NWL  INS  REP  ^Z
```

1.4 ATOK6を組み込む

Quick BASICは、このままでは日本語の入力ができません。

そのために、「一太郎」の日本語フロントエンドプロセッサであるATOK6を使えるようにします。

1. 「一太郎」を起動して、終了させます。
2. ドライブAに、「一太郎」付属のユーティリティディスクを入れます。
3. ドライブBに、Quick BASICのワークディスクを入れます。
4. キーボードから、ATUT［リターン］と打ち込みます。画面上に、ATOK6の環境登録ユーティリティが表示されます。カーソル移動キーを使って、項目の設定を行ないます。

設定の内容は、辞書、外字ファイルをドライブB、日本語の入力位置をQuick BASICのカーソルのある位置（エコー）としました。書き換えるCONFIG.SYSは、ドライブBのワークディスクですから、Bを指定して、［リターン］キーを押します。

付録2 マウスドライバの利用方法

Quick BASIC4.5のマウスドライバには、切り離すことのできる常駐プログラムMOUSE.COMと、CONFIG.SYSにデバイスドライバとして組み込むMOUSE.SYSがあります。

どちらかのファイルを組み込めばマウスが利用できます.

1　MOUSE.COM の組み込み

MOUSE.COM は、常駐型のマウスドライバで、マウスインターフェイスにマウスをつないで利用する場合（バスマウスの場合）は、

```
MOUSE [リターン]
```

で使います。また、RS-232C用のマウス（シリアルマウス）の場合は、

```
MOUSE /C [リターン]
```

で、使うことができます。Quick BASICを起動中に、MOUSE.COMを切り離すことができ、本体メモリが不足気味のときには、とても便利なマウスドライバです。
MOUSE.COMの解放は、

```
MOUSE OFF [リターン]
```

です。起動時には、MOUSE.COMをAUTOEXEC.BATやほかのバッチファイルで起動、解放を行なうことができます。

2　MOUSE.SYS の組み込み

MOUSE.SYS は、CONFIG.SYSに設定するデバイスドライバです。設定の方法は、次のようになります。2つを同時に使うことはできません。

```
バスマウス：      DEVICE=MOUSE.SYS
シリアルマウス：DEVICE=MOUSE.SYS /C
```

付録 3

変数名に使うことができないBASICの予約語

全部で、238語あります。

A	CIRCLE	DEFSNG	FOR	KILL	MKDIR	POINT
ABS	CLEAR	DEFSTR	FRE	KLEN	MKDMBF$	POKE
ACCESS	CLNG	DIM	FREEFILE	KMID$	MKI$	POS
ALIAS	CLOSE	DO	FUNCTION	KPOS$	MKL$	PRESET
AND	CLS	DOUBLE	G	KTN$	MKS$	PRINT
ANY	COLOR	DRAW	GET	L	MKSMBF$	PSET
APPEND	COM	E	GOSUB	LBOUND	MOD	PUT
AS	COMMAND$	ELSE	GOTO	LCASE$	N	R
ASC	COMMON	ELSFIF	H·J	LEFT$	NAME	RANDOM
ATN	CONST	END	HEX$	LEN	NEXT	RANDOMIZE
B	COS	ENDIF	JIS$	LET	NOT	READ
BASE	CSNG	ENVIRON	I	LINE	O	REDIM
BEEP	CSNG$	ENVIRON$	IF	LIST	OCT$	REM
BINARY	CSRLIN	EOF	IMP	LOC	OFF	RESET
BLOAD	CVD	EQV	INKEY$	LOCAL	ON	RESTORE
BSAVE	CVDMBF	ERASE	INP	LOCATE	OPEN	RESUME
BYBAL	CVI	ERDEV	INPUT	LOCK	OPTION	RETURN
C	CVL	ERDEV$	INPUT$	LOF	OR	RIGHT$
CALL	CVS	ERL	INPUT\	LOG	OUT	RMDIR
CALLS	CVSMBF	ERR	INSTER	LONG	OUTPUT	RND
CASE	D	ERROR	INT	LOOP	P	RESET
CDBL	DATA	EXIT	INTEGER	LPOS	PAINT	RTRIM$
CDBL$	DATE$	EXP	IOCTL	LPRINT	PALETTE	RUN
CDECL	DECLARE	F	IOCTL$	LSET	PCOPY	S
CHAIN	DEF	FIELD	IS	LTRIM$	PEEK	SADD
CHDIR	DEFDBL	FILEATTR	K	M	PEN	SCREEN
CHR$	DEFINT	FILES	KEXT$	MID$	PLAY	SEEK
CINT	DEFLNG	FIX	KEY	MKD$	PMAP	SEG

SELECT	SLEEP	STOP	T	TRON	USING	WAIT
SETMEM	SOUND	STR$	TAB	TYPE	V	WEND
SGN	SPACE$	STRIG	TAN	U	VAL	WHILLE
SHARED	SPC	STRING	THEN	UBOUND	VARPTR	WIDTH
SHELL	SQR	STRING$	TIME$	UCASE$	VARPTR$	WINDOW
SIGUNAL	STATIC	SUB	TIMER	UEVENT	VARSEG	WRITE
SIN	STEP	SWAP	TO	UNLOCK	VIEW	X
SINGLE	STICK	SYSTEM	TROFF	UNTIL	W	XOR

付録 4 *V4.5の制限*

1　グラフィックスのパレットモードについて

Quick BASICのパレットモードは、機種によって制限があり、グラフィックスのおまじない、SCREEN文は次のように制限されます。

1．8色中8色モード

SCREEN 0のみ……E/F/U/VM2/VF2/UV2/VX2

2．4096色中16色モード

SCREEN 0、1、2、3……上記以外の機種

2　PC-9801用にカラーコードを変換する変数

PC-9801では、カラーコードとパレット色が違うため、98本来のパレット色を指定できる変数QBGRPNECが用意されています。ただし、文字の色は、Quick BASICのカラーコードで表示されます。

カラーコード	0	1	2	3	4	5	6	7
QuickBASIC のパレット	黒	青	緑	水	赤	紫	黄	白
PC-9801 本来のパレット	黒	青	赤	紫	緑	黄	水	白

MS-DOSのシステムが立ち上がった状態でキーボードから入力するか、AUTOEXEC.BATの中に書き込んでおきます。

SET QBGRPNEC=(任意の文字列：ABCでもXYZでもよい)

1．Quick BASICの起動/終了時にグラフィック画面を消去しない。

2．Quick BASICの起動/終了時にパレットを初期化しない。

付録5　4.2と4.5の違い

1．無印9801では、4.5は使えません。

2．サウンド関係のコマンドが追加されました。

PLAY、SOUND、UEVENT、SLEEP

3．SCREENで[mode]オプションに82が追加されました。

SCREEN82を指定してやることで、テキスト前景色を0にしたときだけ、背景8色を変えることができます。

4．アナログマシンで4096色中8色の表示が可能になりました。

5．WIDTHで20行モードが使えるようになりました（V4.2は、WIDTH 80,25のみ）。

6．V4.2のファイルは読み込めますが、V4.5のファイルはV4.2では読めません。

7．環境変数QBGRPNEC追加で、PC-9801のカラーコードと同じにできます。

付録6　本誌サンプルプログラム内容一覧

第1部　第3章　プログラムのコンパイル

001:¥ITI03¥SPL001.BAS…三角形の面積

002:¥ITI03¥SPL002.BAS…三角形の面積10回繰り返し

003:¥ITI03¥SPL003.BAS…「人と動物」論理演算子

003:¥ITI03¥SPL004.BAS…合計と平均値

004:¥ITI03¥SPL005.BAS…合計と平均値と最大・最小

005:¥ITI03¥SPL006.BAS…数値カウントタイマー

006:¥ITI03¥SPL007.BAS…画面表示カウントタイマー

008:¥ITI03¥SPL008.BAS…物体の落下速度U

009:練習問題１．台形の面積　¥ITI03¥DAIKEI.BAS

010:V4.5コンパイルﾌｧｲﾙ　¥ITI03¥SUM01.EXE 合計と平均値と最大・最小

011:V4.2コンパイルﾌｧｲﾙ　¥ITI03¥SUM02.EXE 合計と平均値と最大・最小

012:V4.2コンパイルﾌｧｲﾙ　¥ITI03¥SUM03.EXE　〃　ﾗﾝﾀｲﾑﾓｼﾞｰﾙがないと走らない

第1部　第4章　表示画面の体裁を整える

001:¥ITI04¥INKORON.BAS…入力・表示のカンマとセミコロン

002:¥ITI04¥PRKORON.BAS…コメントつき表示

003:¥ITI04¥SPL3-001.BAS…LOCATE文

004:¥ITI04¥SPL3-002.BAS…LOCATE文＋マルチステートメント

005:¥ITI04¥SPL3-003.BAS…LOCATE文と変数

006:¥ITI04¥SPL3-004.BAS…Aの一直線表示

007:¥ITI04¥SPL3-005.BAS…Aのランニング

008:¥ITI04¥SPL3-006.BAS…数値変数

009:¥ITI04¥SPL3-007.BAS…変数の型で表示が変わる

010:¥ITI04¥SPL3-008.BAS…表示位置の指定

011:¥ITI04¥SPL3-009.BAS…型の変更

012:¥ITI04¥SPL3-010.BAS…任意の文字数の抽出

013:¥ITI04¥SPL3-011.BAS…入力文字のコード変換

014:¥ITI04¥SPL3-012.BAS…入力文字を右から左へ表示

015:¥ITI04¥SPL3-013.BAS…キャラクタ表示

016:¥ITI04¥SPL3-014.BAS…自作カーソル

第1部　第5章　ファイル処理の基礎知識

001:¥ITI05¥SPL0401.BAS…シーケンシャルファイルデータ入力

002:¥ITI05¥SPL0402.BAS…シーケンシャルファイルデータ読み出し

003:¥ITI05¥SPL0403.BAS…ロータス1-2-3用データ入力

004:¥ITI05¥SPL0403.EXE…ロータス1-2-3用データ入力実行ファイル

005:¥ITI05¥SPL0404.BAS…おまけ：2進数換算プログラム

第2部　第1章 図形を描くための基本的な命令

001:¥NIB¥GR-PR001.BAS…点を表示

002:¥NIB¥GR-PR002.BAS…点を消去

003:¥NIB¥GR-PR003.BAS…画面を横に走る線

004:¥NIB¥GR-PR004.BAS…画面を縦に走る線

005:¥NIB¥GR-PR005.BAS…2点表示

006:¥NIB¥GR-PR006.BAS…斜め線表示

007:¥NIB¥GR-PR007.BAS…相対位置決め

008:¥NIB¥GR-PR008.BAS…四角形

009:¥NIB¥GR-PR009.BAS…色つき四角形

010:¥NIB¥GR-PR010.BAS…円

011:¥NIB¥GR-PR011.BAS…カラーコード線表示

012:¥NIB¥GR-PR012.BAS…カラーコード面表示

013:¥NIB¥GR-PR013.BAS…色の混合

014:¥NIB¥GR-PR014.BAS…四角形と円の移動

第2部　第2章　直線と四角形

001:¥NIB02¥GR-201.BAS…線のスタイル

002:¥NIB02¥GR-202.BAS…グラフィックス基準メッシュ

003:¥NIB02¥GR-203.BAS…乱数による点の表示
004:¥NIB02¥GR-204.BAS…乱数による線の表示
005:¥NIB02¥GR-205.BAS…LINE文を使った三角形
006:¥NIB02¥GR-206.BAS…STEP文を使った三角形
007:¥NIB02¥GR-207.BAS…READ～DATAの三角形
008:¥NIB02¥GR-208.BAS…円の乱舞
009:¥NIB02¥GR-209.BAS…四角の乱舞
010:¥NIB02¥GR-210.BAS…楕円の乱舞
011:¥NIB02¥GR-211.BAS…乱数を使った色の発生
012:¥NIB02¥GR-212.BAS…PUT、GETでウインドウを作る

第２部　第３章　曲線

001:¥NIBU03¥PR301.BAS…角度とラジアン
002:¥NIBU03¥PR302.BAS…扇形
003:¥NIBU03¥PR303.BAS…連続した扇形
004:¥NIBU03¥PR304.BAS…二次関数の曲線
005:¥NIBU03¥PR305.BAS…連続した二次関数の曲線
006:¥NIBU03¥PR306.BAS…三次関数の曲線
007:¥NIBU03¥PR307.BAS…サインカーブ
008:¥NIBU03¥PR308.BAS…円のサイカーブ
009:¥NIBU03¥PR309.BAS…リザージュ曲線
010:¥NIBU03¥PR310.BAS…一本の花びら
011:¥NIBU03¥PR311.BAS…複数の花びら
012:¥NIBU03¥PR312.BAS…3次元関数の応用（書籍未掲載）
013:¥NIBU03¥PR313.BAS…サインカーブの応用（書籍未掲載）

第２部　第４章　図形の回転

001:¥NIBU04¥PR401.BAS…扇形の30度毎の表示
002:¥NIBU04¥PR402.BAS…カラードーナツ
003:¥NIBU04¥PR403.BAS…地下鉄工事
004:¥NIBU04¥PR404.BAS…直線を使った曲線＝１＝

005:¥NIBU04¥PR405.BAS…直線を使った曲線＝2＝

006:¥NIBU04¥PR406.BAS…直線を使った曲線＝3＝

007:¥NIBU04¥PR407.BAS…直線を使った曲線＝4＝

008:¥NIBU04¥PR408.BAS…直線を使った曲線＝5＝

009:¥NIBU04¥PR409.BAS…多角形を作る

010:¥NIBU04¥PR410.BAS…多角錘

011:¥NIBU04¥PR411.BAS…階乗式を使った直線

012:¥NIBU04¥PR412.BAS…息をする四角形

013:¥NIBU04¥PR413.BAS…中三針アナログ時計

第2部　第5章　N88-BASICからの移植

001:¥NIB05¥QUAD.BAS…赤松氏プログラムをMS-DOSに変換（走らない）

002:¥NIB05¥PR501.BAS…赤松氏プログラムをQB用にデバッグ

003:¥NIB05¥PR502.BAS…赤松氏プログラムの改良

004:¥NIB05¥SAKUHYO.BAS…N88-BASIC（MS-DOS用)作表プログラム

005:¥NIB05¥PR503.BAS…作表プログラムをQB用にデバッグ

006:¥NIB05¥PR504.BAS…自作カーソルとワープゾーン

読者の皆さまにお願い

本書をお読みくださいましたあなたのご感想はいかがでございましたでしょうか……。

私たち『日本文芸社』は常に、あらゆる人びとに愛され、親しまれ、そして何らかの指標になり役立つ本でありたいということを願って、編集に心をこめておりますが、この上とも皆さまとともに歩むため、ご希望なり、ご意見なり、またお読みになりたい内容のものがございましたら、左記あてにどしどしお寄せくださるようお願いいたします。

なお、誤植や乱丁のないように努めておりますが、もし誤りの個所にお気づきになりましたら、ご職業、年齢などをお書き添えのうえご教示くだされば、まことに幸甚と存じます。

東京都千代田区神田神保町一ノ八
日本文芸社

Quick BASIC 初歩の初歩講座

著　者　遠藤謙一
発行者　兵頭武郎
印刷所　長苗印刷株式会社
製本所　(有)松本紙工

発行所　東京都千代田区神田神保町１－８
株式会社　日本文芸社
振替口座　東京　8-73081番
電話　(代表) 03-294-8931～6

落丁本・乱丁本はお取り替え致します　担当：松坂
Printed in Japan
6016-1485
ISBN4-537-01467-9 C0055
110900808-110900808N01

Quick BASIC 初歩の初歩講座

定価1,600円(本体1,553円)

平成2年8月8日発行

著　者　遠 藤 謙 一

発行者　兵 頭 武 郎

発行所　株式会社　日本文芸社

〒101　東京都千代田区神田神保町1の8

TEL代表(03)294-8931

振替口座　東京8-73081

落丁・乱丁はおとりかえいたします

Quick BASIC
初歩の初歩講座

興味深いショート・プログラムを豊富に収録。
そのサンプル・プログラムをもとに、プログラムの組み方を初め、
コンパイルやリンカーの使い方を徹底的にわかりやすく解説。

ASIC
uick

遠藤 謙一

定価1600円（本体1553円）

6016-1485
ISBN4-537-01467-9 C0055 P1600E

日本文芸社